DOCUMENT RESUME

ED 267 444 CS 209 634

AUTHOR Gallagher, Brian
TITLE Microcomputers and Word Processing Programs: An Evaluation and Critique. Research Monograph Series Report No. 9.
INSTITUTION City Univ. of New York, N.Y. Office of Academic Affairs.
PUB DATE Mar 85
NOTE 196p.
PUB TYPE Guides - Non-Classroom Use (055) -- Reports - Evaluative/Feasibility (142)

EDRS PRICE MF01/PC08 Plus Postage.
DESCRIPTORS *Computer Software; *Courseware; Evaluation Criteria; Higher Education; *Microcomputers; Program Descriptions; *Program Evaluation; Teacher Attitudes; *Word Processing; *Writing Instruction

ABSTRACT

The use of word processing as an instructional tool is the focus of this monograph. The introductory section provides an overview of word processing and its use in higher education, while the second section explores the use of word processing in writing courses, including word processing for basic writers and learning disabled students, word processing and collaborative learning, and teaching the word processor as a writing tool. Word processing for general faculty use is discussed in the third section, which takes into consideration faculty resistance to computers and other factors to consider when choosing a word processing program. The next section contains evaluations of the following word processing programs: Personal QWERTY, Wordvision, Megawriter, Textra; PFS:Write, Select Write, Volkswriter Deluxe, Officewriter, Easywriter, Superwriter, Peachtext, Perfect Writer, Wordstar, Visiword, Spellbinder, Easywriter II, Wordperfect, Microsoft Word, Powertext, Multimate, and various spelling checkers. The final section examines some of the social, educational, epistemological, and ontological issues surrounding word processing and microcomputers. Five pages of references conclude the report. (HOD)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED267444

RESEARCH MONOGRAPH SERIES
REPORT NO. 9

MICROCOMPUTERS AND WORD PROCESSING PROGRAMS

AN EVALUATION AND CRITIQUE

March 1985

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

X This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

Instructional Resource Center
Office of Academic Affairs
The City University of New York
535 East 80th Street
New York, New York 10021

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Barbara Gonzales

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

CS209634

THE CITY UNIVERSITY OF NEW YORK

BOARD OF TRUSTEES

James P. Murphy, *Chairperson*
Edith B. Everett, *Vice Chairperson*

Paul P. Baard
Blanche Bernstein
Sylvia Bloom
Louis C. Cenci
Armand D'Angelo
Judah Gribetz
William R. Howard
Harold M. Jacobs
Robert L. Polk
Joaquin Rivera
Margaret A. Titone
Brenda Farrow White
Melvin E. Lowe, *ex officio*
Henry Wasser, *ex officio*

Joseph S. Murphy, *Chancellor*

THE COLLEGES

Baruch College—Joel Segall, *President*
Borough of Manhattan Community College—Joshua L. Smith, *President*
Bronx Community College—Roscoe C. Brown, Jr., *President*
Brooklyn College—Robert L. Hess, *President*
City College—Bernard W. Harleston, *President*
City University School of Law at Queens College—Charles R. Halpern, *Dean*
Graduate School and University Center—Harold M. Proshansky, *President*
Hostos Community College—Flora Mancuso Edwards, *President*
Hunter College—Donna E. Shalala, *President*
John Jay College of Criminal Justice—Gerald W. Lynch, *President*
Kingsborough Community College—Leon M. Goldstein, *President*
LaGuardia Community College—Joseph Shenker, *President*
Lehman College—Leonard Lief, *President*
Medgar Evers College—Jay Carrington Chunn, II, *President*
Mount Sinai School of Medicine (affiliated)—James F. Glenn, *President*
New York City Technical College—Ursula C. Schwerin, *President*
Queens College—William Hamovitch, *Acting President*
Queensborough Community College—Kurt R. Schmeller, *President*
The College of Staten Island—Edmond E. Volpe, *President*
York College—Milton G. Bassin, *President*

MICROCOMPUTERS AND WORD PROCESSING PROGRAMS

AN EVALUATION AND CRITIQUE

March 1985

By
Brian Gallagher
English Department
LaGuardia Community College

ACKNOWLEDGEMENTS

A number of persons at the Instructional Resource Center of CUNY's Office of Academic Affairs helped on this project. Dean Michael Ribaudo conceived of it and procured the programs for testing; Dean Marie Jean Lederman helped define the project and the resulting monograph. The main work of editing was done by Barry Kwalick. Ruth Davis worked on the initial draft, and Virginia Slaughter saw the project through its editing and publication stages. I am grateful to each of them. My larger debt to the Instructional Resource Center is indicated by the number of its publications that are cited in the monograph.

I also want to thank my colleagues in the microcomputer/writing program at LaGuardia Community College: Daniel Lynch, Joan Greenbaum, Estelle Schneider, Karen Pearl and, especially, Marian Arkin. Professor Arkin read a draft of the monograph and offered much helpful advice--even more important was her support throughout the project. I thank the Kiewit Computer Center at Dartmouth College, and especially Tom Byrne and Anne Huggins, for use of computer facilities during the summer of 1984. I also owe debts of gratitude to the students I have taught these last two years in the microcomputer/ writing program--they have taught me as much about using word processing and computers in education as I, hopefully, have taught them about writing--and to the New York State Vocational Education grant program which has so generously funded our work at LaGuardia.

© Copyright 1985 by
The Instructional Resource Center, Office of Academic Affairs
The City University of New York

TABLE OF CONTENTS

Page

1. INTRODUCTION
   - 1.1 Prejudices and Parameters..... 1
   - 1.2 Word Processing: An Overview and an Instance..... 2
   - 1.3 Word Processing: Major and Minor Capabilities..... 4
   - 1.4 Computers and Word Processing in Higher Education: Future Prospects and Present Problems..... 10

2. WORD PROCESSING IN WRITING COURSES
   - 2.1 Word Processing, Microcomputers, and Pedagogical Change..... 15
   - 2.2 LaGuardia's Microcomputer/Writing Program: A Summary Description..... 17
   - 2.3 Word Processing for Basic Writers..... 18
   - 2.4 Word Processing for Learning Disabled Students..... 20
   - 2.5 Word Processing and Collaborative Learning..... 21
   - 2.6 Teaching the Word Processor as a Writing Tool..... 22
   - 2.7 Word Processing in the Writing Classroom: Success Factors..... 23
   - 2.8 Word Processing in the Writing Classroom: Special Techniques..... 28
   - 2.9 Word Processing in the Writing Classroom: Real and Potential Problems..... 29
   - 2.10 Other Computer Programs for Writing Courses..... 31

3. WORD PROCESSING FOR GENERAL FACULTY USE
   - 3.1 Faculty Resistance to Computers and Word Processing: A Mild Polemic..... 38
   - 3.2 Writing vs. Recording on the Word Processor..... 41
   - 3.3 Word Processing in Higher Education: Some Effects on the Professoriate..... 41
   - 3.4 Word Processing Programs for Faculty Use: Some Points to Consider When Choosing..... 44

4. EVALUATIONS OF SPECIFIC PROGRAMS
   - 4.1 Principles of Analysis..... 49
   - 4.2 Inexpensive Programs..... 51
     - Personal QWERTY..... 51
     - WORDVISION..... 54
     - MEGAWRITER..... 57
     - TEXTRA..... 59
     - PFS: WRITE..... 63
     - SELECT WRITE..... 66
   - 4.3 Moderately Priced Programs..... 70
     - VOLKSWRITER DELUXE..... 70
     - OFFICEWRITER..... 73
     - EASYWRITER..... 77
     - SUPERWRITER..... 80
     - PEACHTEXT..... 84
     - PERFECT WRITER..... 89

4.4 Expensive Programs..... 93
WORDSTAR..... 94
VISIWORD.....100
SPELLBINDER.....104
EASYWRITER II.....108
WORDPERFECT.....113
MICROSOFT WORD.....119
POWERTEXT.....123
MULTIMATE.....129
4.5 Spelling Checkers.....135

5. WORD PROCESSING AND MICROCOMPUTERS: SOME SOCIAL, EDUCATIONAL, EPISTEMOLOGICAL, AND ONTOLOGICAL ISSUES
5.1 General Influences on Writing.....140
5.2 The Future of Word Processing and Microcomputers in Education.....145
5.3 Writing as an Electronic Act.....151
5.4 Word Processing and Pseudo-Consciousness.....154
5.5 Word Processing and the Ontology of Writing.....158
5.6 Word Processing and the Idea of a "Text".....161

APPENDIX A:
Sample Student Direction Sheet.....167

APPENDIX B:
Class Diaries.....171

WORKS CITED.....187

CHAPTER 1

INTRODUCTION

1.1 Prejudices and Parameters

One can hardly fail to notice it: microcomputers have become a very big, and often a very confusing, business. There are presently over two hundred types of personal computers on the market, at least that many word processing programs, and perhaps 40,000 computer programs available overall (Gonzales xii). Even "Charlie Chaplin," once a symbol of the "little fellow" in all his quaint and shabby individuality, has gone over to pushing these machines for the largest computer corporation in the world.

Despite all the hoopla, there is still a fair chance that as a teacher, scholar and/or writer you may not really need a personal computer, at least not badly enough to justify its considerable expense. There is, however, a better chance that a personal computer is a device that could serve you well in several limited but important respects--most particularly because it could save you significant amounts of time.[1] There is an even greater chance that as a teacher of writing, or many other subjects, you will simply have to learn the basics of microcomputer use to function in the classroom in coming years.[2]

Presently, computer books outsell computers at a 10:1 ratio, and the reason is not simply that books are cheap and computers expensive. There is obviously something titillating, vaguely "pornographic," in reading about the acrobatic machinations and lightning-quick "interfaces" of the microcomputer, actions one can easily fantasize performing in the privacy of one's electronic home. All too often the excitement engendered by ever-present computer advertising carries over into books written about computers (hardware) and their programs (software). Seemingly, a real computer maven, despite common assurances that his (almost always "his") book or article is a "basic" guide for "those who know nothing about computers," cannot resist hopping from one topic to another with sometimes bewildering speed. No sooner does the reader begin to get a sense of how a particular kind of program might actually work, than the writer is off extolling alternate hardware combinations to run it, suggesting peripheral programs to add to it, and revealing secrets that can make work just a few seconds faster.

This monograph, contrariwise, concentrates almost exclusively on one aspect of microcomputer use: word processing programs. There is a great deal to say on this topic alone, including some speculative points to be made about the likely effects word processing will have on writing, on our notion of a "text," and on the teaching of writing and other subjects over the next few decades. Other writing-related programs, including computer-assisted instruction (CAI), are discussed briefly at the end of the second chapter. Programs to check spelling, which are sometimes part of word processing programs (although they usually cost extra), are mentioned at the end of the fourth chapter.

Hardware is not discussed at all. To simplify matters, all the word processing programs analyzed in this monograph are programs written or adapted for the IBM personal computer, the most popular personal computer on the market. Many of these programs can also be run on other personal computers, especially Apples--and the rapid adaptation of programs for the more popular microcomputers is an ongoing process. For example, Microsoft Word is presently (June 1984) being rewritten for Apple's new Macintosh computer and should be available by the time you read this. Rather than try to list all the machines on which the word processing programs analyzed here can run, I have provided the name, address, and telephone number of the distributor of each, and suggest you contact the producer to see if the program that interests you will run on the equipment you have, or expect to have, available.

Part of the information that follows, particularly in the first chapter, will seem rudimentary for those acquainted in some degree with microcomputers and their use as word processors. However, it is my impression--and I have confirmed it with a number of colleagues who have used word processing in the classroom and for other academic activities--that most academics still know almost nothing about word processing and its potential uses for them as teachers, scholars, and writers. To include this group, the most basic facts and attributes of word processing programs are explained in some detail below. Those more knowledgeable about word processing will still, I trust, find some points to ponder in the last section of this chapter and in chapters 2, 4 and, especially, 5.

1.2 Word Processing: An Overview and an Instance

Word processing is a very efficient electronic system for recording, formatting, editing, storing, retrieving, combining, and printing documents. The key difference between a word processed document and one produced by even the fanciest electronic typewriter is that the entire word processed document is first and foremost an electronic entity. As such, it is always and everywhere subject to instantaneous alteration. The production of a material text--i.e., characters imprinted on a piece of paper--is very much a discrete act, a material realization of the latest electronically created version of the document. With a word processor,[3] one is never limited by the relative immutability of words in a typewritten or handwritten document. Even in draft versions, these will allow only a certain amount of correction, revision, and alteration; then they must be retyped or rewritten. With a word processor, however, one need type only a single version of any document. Insertions are merely typed in at the chosen spot, with the following text being automatically pushed ahead to make room for them; deletions, from a single character to whole pages, remove the designated portion of text entirely and automatically close the gap by moving up the text. Moreover, one can stop the reworking process at any point, give the command to print all or part of the document, and have the piece in hand, through the very last change, in a matter of minutes. It is always possible, therefore, to judge just how much revisions have changed the document.

The substantial benefits of word processing can be appreciated by imagining oneself in the following, not untypical, academic situation: you have just typed the fourth and what you hope will be the last version of a thirty-page article you have been working on for several months. While

proofing this copy, you locate ten typos, four spelling errors, and realize that two pages are numbered "16," meaning that subsequent pagination is wrong. In addition, you find one sentence you would prefer to eliminate, an unnecessary footnote, a point that needs further elaboration, and two phrases that could be made smoother. Unfortunately, making all these changes would mean retyping the essay from the fourth page forward--and, of course, retyping so many pages might introduce a whole new set of errors. Most likely, you would make those small corrections (typos, misspellings, pagination, perhaps some rewordings) that could be effected without major retyping, trusting that the remaining imperfections would not be significant enough to influence the main fate of the piece.

Were this document on a word processor, all the desired changes could be made in a third the time it would take to make only the minimal changes on the typed text. Like savings in time and similar chances to include every desired change would have occurred at every stage had the text been written on the computer. Several errors might not have occurred at all: pagination can easily be made automatic in almost all programs; WordPerfect, PowerText, and several other programs allow for automatic footnote placement at the bottom of the page, with automatic renumbering and repositioning of footnotes whenever one is added or deleted; using EasySpeller with EasyWriter II would have permitted the spelling of any word to be verified, in only two keystrokes, directly after it was typed.

It is not entirely accurate, as the naysayers would have it, that word processing in no way makes one a better writer. Certainly, writing on a computer will not magically or instantly provide anyone with ideas, a sense of organization, an effective and individual style, or a grasp of efficacious revision strategies, but to the extent such things are influenced by the ease and speed of the writing and revising, the legibility of each successive draft, and the knowledge that changes are always possible and easily effected, then writing on a word processor does help improve one's writing. For those who do not know how to type--or for those who, like myself, have worked out a quirky, and not particularly efficient, hunt-and-peck system--word processing is a decided boon. Text needs only be typed once, and typing it is faster (because there is no worry over committing errors to paper) and somewhat easier (since the machine automatically handles carriage returns, page breaks, and the like). For all writers, the fact that no draft after the first has to be typed in anything close to its entirety makes word processing, simply but significantly, a means of writing more quickly and, thus, of writing more. Ideally, word processing should also, as William Zinsser puts it, "greatly help people to clean up their sentences by focussing their mind on the act of writing and revising"(23).

As the authors of one of the most innovative composition textbooks of recent years observe, "The way to write better is to write more--but not alone, not aimlessly, not without guidance and encouragement" (Scholes and Comley vii). Any teacher or administrator working to incorporate word processing in writing or other courses would do well to bear the whole of this observation in mind. Certainly, word processing will help students to write more and to write more easily, but, as we shall see in the second chapter, a battery of other proven pedagogical techniques must be integrated with work on the word processor. For now, we might just note that writing on computers

goes a fair way toward obviating some of the common problems student writers, particularly beginning student writers, encounter:

> Initial writing blocks are overcome when working with a screen terminal and its ease of erasing through simple backspacing, a capability that often stimulates a flow of words that paper inhibits. Mechanical difficulties with poor handwriting and spelling are overcome when composing on a computer; the copy looks professional, boosting the writer's self-confidence; spelling is easily corrected after a first draft, causing writers to be more relaxed about putting words down. Because of the ease of revising without tedious recopying, writers are more open to suggestions for change and are less inhibited about implementing suggestions. (Standiford et al. 23)

By itself, word processing is far from being a total solution to the problem of teaching students to write clear, effective, well-organized prose, but it is obviously one of the most valuable aids in that process.

It is not without point that the mathematician who designed the first computer capable of being programmed, the Hungarian-American John von Neumann, "is perhaps even better known for his contribution to the theory of games" (Ayer 15).[4] For all writers and especially for student writers, the ability to "play" with one's text on the computer is a large part of the attraction of writing on the machine--and if such play is not allowed to become an end in itself, it can be a vital extension of the sense of language play that is at the core of all written (and verbal) communication.

On the other hand, it should also be remembered that word processing was first developed as, and essentially remains, a business instrument. Whatever use teachers, students, and writers find for it will be something of an adaptation, a function ancillary to word processing's primary task of producing business documents. A common theme in much recent academic writing about computers is that they can and must be used "humanistically," rather than as the means for self-aggrandizement or money-making.[5] It would be more honest, I believe, to admit that computers are, effectively if not literally, "machines" that can and will improve our ability to earn our academic keep as teachers, administrators, scholars, and writers. The overall importance microcomputers will assume in education over the next few decades is still an open question. Nonetheless, because word processing is so clearly cost-effective, it is quickly becoming virtually the only regular way of producing documents--both without and within academia.

1.3 Word Processing: Major and Minor Capabilities

Using a word processing program demands no general knowledge of how to operate a computer, but only the ability to follow the instructions of the accompanying documentation (manual, on-screen menu, tutorial, etc.). Admittedly, reading instructions sometimes requires the reduction of wordy, oblique, obscure, and inessential explanations to a series of simple, clear steps, but anyone approaching the task with an open mind should be able to master most programs in five hours and be able to work very comfortably with them in ten. Be warned, though: the first few hours can be very trying, even

disheartening, with text suddenly vanishing, formats inexplicably changing, mysterious messages rapidly appearing and disappearing--which is precisely why an open mind is required. After two hours you may never want to write on a word processor again; after five you may never want to write on anything else.

Although the more sophisticated programs allow for several hundred different editing and formatting functions (see Chapter 4), the major capabilities of word processing are few and are common to all programs, although the ease and convenience of performing them can vary markedly. Most of the advantages of word processing derive from the way it handles the following operations:

1) Insertions. With some programs, you are in the "insert mode" as you write; with others, you must give a command to enter it. A number of programs clear a space of several lines to make room for insertions, while others simply push the text ahead one space at a time as the insertion is typed. Often, the program must be returned to the "write mode" when the insertion is completed, and some programs require a command to reformat the text to specified line lengths after insertions (or deletions). Yet, whatever the specific steps, all programs permit the easy insertion of any amount of material and the expansion of the document to accommodate it.

2) Deletions. Programs can perform deletions up to eight ways: by space, by word, by line, by sentence, by column, by paragraph, by page, and by designated block. Many programs also activate the "destructive backspace" key (found on IBM and other computers), which moves the cursor (the box or blinking light that signifies where you are in the text) back one space and eliminates whatever was in that space--a very useful feature to correct errant keystrokes as you type. Naturally, whole files (entire documents) can also be deleted, and most programs will, wisely, ask you to confirm a "delete file" command before undertaking such a large-scale removal. A number of programs contain a "disk buffer," which stores the most recently deleted section of text and allows it to be inserted again if an error has been made or if you simply change your mind about the deletion. At the very least, all programs permit the swift deletion of any-sized units and automatically join the text following the deletion to that preceding it.

3) Word wrap and page breaks. With word processing programs you do not use the "carriage return," unless you want to end a paragraph or leave empty space at the end of a line, since words will automatically be moved down to the next line when they go past the margin. Almost all programs have a "justification" feature which can be turned off and on; when on, it will automatically create a flush right margin by inserting extra empty spaces in each line. Turning it off will give you a "ragged right" margin. In either case, the word-wrap feature is always working. It may, to be sure, take an experienced typist some time to break the habit of hitting the carriage return at the end of a line, but once you adjust to this feature, typing becomes simply a matter of entering the words and letting the computer place and align them. Many programs also have a "soft hyphenation" function, which allows for the hyphenation of words at the end of lines, but which removes the hyphen and rejoins the word automatically should a revision of the text shift a hyphenated word from its end-of-line position.

All programs are preset to break the text into standard page length divisions (54 or 55 lines per page), after which a two-inch space (1" for the bottom margin of the first page, 1" for the top margin of the second) is inserted as pages are printed. This can be reset to accommodate pages of virtually any length--or turned off to create a "continuous format" printout. Some programs also offer "widow/orphan protection" (word processing terminology can be melodramatic), which means that you can order the program never to put the first line of a paragraph as the last line on a page or the last line of a paragraph as the first line on a page.

Word wrap, automatic page breaks, and allied features allow the writer to type in text without the distraction of having to attend to line and page lengths and the overall "look" of the document. Even if the document does not appear right after it is printed, its layout can be altered and a new version printed in just a few minutes.

4) Block moves and copies. All programs provide means of designating blocks of text of virtually any length which can then be moved to any other place in the document, or even to another document. This "electronic cut and paste" system is extremely helpful for writers who prefer to write out their material first and leave the explicit organization of it for later drafts. All programs also have a copy function, which works the same way as the move function, except that the designated block also remains where it was as well as being inserted elsewhere--a feature that is most useful for moving text to another document without destroying the integrity of the original. Because the whole word processed document is dynamic, sections of it can be moved, rearranged, and copied in a matter of seconds.

5) Multiple drafts. It is possible to keep every draft of a document on the computer, for cross-referencing and possible revival (in whole or part). Therefore, writers who use the move function extensively need not worry about restructuring their documents overmuch, since they can always return to earlier versions of them. Naturally, writers who work slowly and revise definitively and do not want the distraction of alternate versions can have each subsequent revision replace its predecessor. Still, any writer might benefit from once or twice keeping every draft of a piece and printing out neat, legible copies of each for review and analysis.

Some programs make storing each draft very easy--e.g., when you are ready to "save" (record your document permanently on a disk) in VisiWord, you must first indicate under what name you want the document saved; giving it a slightly different name (say, "Draft B" rather than "Draft A") will save it and keep the previous version as well. Other programs require some forethought if you want to preserve every draft--e.g., with WordStar you need to create a second copy of the draft you want to rework, then rename that draft, before you begin your alterations. (These procedures take a minute or so, but are not difficult to perform.) In any case, there is always a relatively simple way to preserve and recall any and all drafts of a document.

6) Automatic format creation (and format storage). All word processing programs divide functions between two phases--the editing phase and the formatting phase (although, in fact, all programs relegate some format functions to the editing phase). Editing functions are those that are enacted as the text is entered and revised; formatting functions are those enacted

when the text is printed. "Screen-oriented" programs will enact many functions in the editing phase; thus, if you tell WordStar you want triple-spacing, your text will appear triple-spaced on the screen. "Print-oriented" programs leave the enactment of many functions for the print phase--e.g., if you order PowerText to triple-space, your command ("/SPA [3] <cr>") will appear on the screen, but your text will still be single-spaced there.

Both approaches have their advantages. Screen-oriented programs allow you to see your text just about (although not exactly) the way it will appear when printed, making needed adjustments and possible reshapings easier to spot and do early on in your typing. Print-oriented programs always allow you to see twenty-four lines of text, no matter what the spacing will be at printing, and have more editing and formatting designations on the screen, which sometimes makes changing layouts easier and quicker. With highly screen-oriented programs, the specifications for each document--margins, columns, spacing, alternate page lengths, justification on/off, etc.--must be created with the document, although this is not a particularly difficult process. With print-oriented programs, you can often, as with VisiWord, store standard formats and simply invoke the appropriate one at the time of printing. Many print-oriented programs also have a "print-to-screen" capability, which puts the whole document on the screen as it will look when printed, "scrolling it up" and permitting you to freeze it at any point to check and note things you might want to change. With both types of programs, however, it is a simple matter to create and alter document formats.

7) Boilerplating and mailmerging. Even the simplest programs allow for the easy creation of personalized form letters and like documents. One could, for instance, type the sender's address, body, text, and complementary closing of a letter, then copy that material to a new file, where only the inside address, salutation, and other personalizing insertions would need be added--and so again and again. Alternately, one might type the full letter to the first addressee, print it, and then erase the information relating to the first addressee, replace it with that relating to the second, print that version--and so on.

Such work will be even easier if you use a mailmerge program like EasyMailer or WordStar's MailMerge. With these you merely type your form letter and a list of addresses, and the program automatically merges the variable information with the standardized material. Such programs allow for very intricate tailoring of standardized documents to accommodate individualized entries in many places. Whether a mailmerge program is used or advantage is simply taken of any program's ability to move and copy text instantly, word processing will make the creation of form letters and similar documents a much quicker, less tedious, and more individualized process.

All these major advantages of word processing do not come without some small allied disadvantages. For example, one can at best see twenty-four lines of text on the screen at any time. Text can be "scrolled" up or down a line at a time, and most programs allow the document to be searched backward or forward a screen at a time. Many also have commands that will move to the beginning or end of a document, or to any one of a number of preset "markers" within it. Still, it is much easier to flip back through a pile of

typewritten sheets when you are looking for something you remember saying ten or so pages ago.

A more significant disadvantage is that it is difficult and inefficient to check a point in one document while working on another, since doing so often involves several openings and closings of files (and perhaps locating and activating another disk). Most writers will prefer to work with printed versions of any supplementary documents, thereby obviating the word processor's supposed advantage of storing documents electronically and so eliminating the clutter of paper. However, the fact that only a single printout of a document normally need be kept around, since more copies can always be printed as needed, does at least save on paper storage space.

A different, and potentially more serious, kind of disadvantage is that word processing can encourage a lack of closure in writing. With alterations always and easily possible, it is a temptation to keep revising and refining a document forever, never coming to a satisfactory end point. Rumor has it that one highly respected American novelist finished a long-awaited novel several years ago, but refuses to surrender it to his publisher because he is still tinkering with it on his word processor. Occasionally, students will hold on to their word processed papers with similar tenacity.

The rote operation of the word processor, which is the prime source of its usefulness, also makes it less than ideal for certain tasks. Carl Townsend lists three fairly common situations where a word processor is less useful than a typewriter:

1. The typing of one-of-a-kind documents in which very little text is repeated between documents.

2. Entering data to forms using a typewriter (it should be noted, however, that the form entry may not be necessary if an "electronic" form can be created on a computer or word-processing system).

3. Creating camera-ready copy using a lot of cutting and pasting of different type styles and illustrations. (6)

Word processors may have made typewriters obsolete, but they have not yet made them entirely dispensable.

Although the observation is implicit in much of what has already been said, it is worth noting explicitly that all functions normally performed manually on a typewriter, like setting margins and tabs, can be performed just as easily on a word processor--sometimes more easily. For example, creating superscripts and subscripts is typically done by typing two command keystrokes, then the number(s), then two more command keystrokes--a procedure roughly equivalent to turning the typewriter roller up or down half a line, typing the number(s), and repositioning the roller. Some "manual" functions can be performed more quickly on a word processor, although the time saved is too negligible to be the significant benefit computer enthusiasts would have you believe it to be. For instance, if you want to doublestrike a sentence for emphasis, you would not need to strike every character twice, but only to

turn the doublestrike function on before the first character and off after the last. Many programs also offer boldface, which restrikes a character a fraction of an inch to the right, thereby making an even darker impression than doublestriking. At least one "manual" function, centering, is much more efficiently done on a word processor. Normally, one or two keystrokes will center the text on any line, although all programs would do well to emulate those that easily permit centering after text is typed and not just as it is being typed.

The minor capabilities of word processing are many and vary from program to program. The number and nature of the additional features your writing requires will determine which program will suit you best. Generally speaking, the closer your documents need to resemble standard printed formats and layouts, the more sophisticated the program you will need--and the more you will have to pay to get it. For example, only some programs support on-page footnotes, double-column printing, and interface with printers capable of proportional spacing (e.g., allotting three times the space for "m" as for "i").

Other, more common, minor capabilities can be occasionally helpful, like the capacity to write small parts of a long document as separate files (for easy handling) and then "link" them together when the document is completed. Some much vaunted features, like "search and replace," can be useful for some writers. All programs can locate a word or string of words everywhere it appears in a document and, if desired, replace it with a designated alternate, either all at once or on a case-by-case basis, where the user is given the option of the substitution. This feature is also potentially useful for some stylistic analyses, since it will enable you very rapidly to locate every usage of a word or locution in a document. Of course, when you discover that you have consistently misspelled a word, you can simply order the machine to locate every wrong spelling and replace it with the right one.

The more powerful programs offer "spooling," which simply means that you can work on one document while another is being printed--although the stopping and going of the printer and the gap between the typing of some material and its appearance on the screen (the printer and terminal are "time-sharing," with only one actually working in any given minisecond) may prove too much of a distraction to make this a practical procedure. Many designers of computer programs, to judge by the manuals that accompany them, are not cognizant of the fact that writers, and even typists, need certain enforced breaks, like those provided while a word processor is printing out.

There are, to be sure, other functions in various programs that users will find more or less helpful, depending on their writing habits and writing situations. A good many of these features are covered in the analysis of specific programs in Chapter 4, with the intervening chapters looking at the rather different demands made of word processing in writing and other classrooms and by the more varied work of teachers-writers. Since all three of these chapters proceed on the assumption that computers in general and word processing in particular are entering the academy full force, we might first briefly consider some of the larger implications of this revolution in higher and, indeed, all education.

1.4 Computers and Word Processing in Higher Education: Future Prospects and Present Problems

The City University of New York now has a "University Dean for Academic Computing" functioning as part of its central office. A professor in Texas has recently set up a database "English Microlab Registry." The Council of Writing Program Administrators is one of a number of groups presently conducting a nationwide survey of computer use at colleges and universities. One of the key questions on its questionnaire--"Does your school have a computer literacy requirement?"--is an issue the curriculum committee at my own college has been debating heatedly, but as yet inconclusively, for over a year. In short, I am constantly "encountering" the computer at my institution and in my field; no doubt much the same is true nowadays for almost any academic in almost any field.

Whether computers, in particular microcomputers, eventually prove to be the promised boon to higher education or just more or less useful educational aids, it is clear that computers will not fail to achieve their educational promise through lack of opportunity. Given the coming saturation of college and university curricula with computer-related courses, most faculty will have to learn something about their use, if only to make an informed decision on how and why computers cannot be used in their own pedagogical, administrative, and scholarly work.

The situation of the computer in the writing classroom reflects, in miniature, the whole, somewhat problematical, position of computers in education. Even the best pieces on the use of computers in teaching writing often still reflect too great a degree of enthusiasm. For instance, the Director of Studies at the Groton School avers, "I have found the word processor to be perhaps the most powerful tool I have yet encountered in the teaching of composition" (Gula 31). And one of the leading developers of computer-aided instruction for writing, Hugh Burns, trumpets "Invention as Electronic Orchestration," a concept that would be less worrisome were its elements reversed, since "electronic orchestration as invention" at least suggests the freedom to use or not use the computer for this very necessary and very difficult step in the writing process.

In a few years, teachers of writing in higher education may have to face the word processor in another way. As Zinsser observes, "The new technology could erase the biggest obstacle that frustrates children learning to write: the sheer labor of writing"(110). If that "could" proves to be a "does," then teachers at the latter end of the educational system will have to adjust to instructing many students for whom the word processor was the first writing implement and for whom it remains the most essential one. And if, as some experts predict, voice-activated word processors become a practical reality within the next decade, educators at all levels will have to make the even larger adjustment of reshaping pedagogy to handle the virtual obliteration of the long-standing distinction between oral and written discourse.

For now though, we might recur to the very basic but often ignored truism that to make use of a computer one must have access to it. Not surprisingly, it is usually the richer schools, at all levels, which have the largest and best computer and word processing facilities. Ironically, many times it is those students whose families can easily afford (or already possess) a

personal computer who have ready and free access to institutional computer resources. Poorer institutions typically not only have scanter computer resources, but they also have many more students who must depend on institutional computer resources or go without. The difference in access to computer facilities, both at home and at the institution is, in fact, rapidly widening the already discernible gap between students who receive a quality education in our society and those who must settle for a "make-do" one. Offering a higher education that involves full-scale involvement with computer resources is becoming an important way of enhancing an institution's scope, prestige, and appeal to prospective students. Recently, two things happened at the already highly respected Clarkson College of Technology: it changed its designation to "Clarkson University," and it began to require all entering students to purchase a personal computer--to be paid for by a $200 addition to each semester's tuition--that would link them to the university's powerful mainframe unit. The two developments are not, we might guess, entirely unrelated.

The National Council of Teachers of English is one organization that has begun to recognize and protest some of the non-egalitarian tendencies of the "computer revolution" in education:

> As computer use spreads, so do computer-related problems. Poor methods and materials flourish with the good. Generous access to computers coexists with no access, a problem sometimes related to the socio-economic level of the students or the school and sometimes related to departmental divisions within the school. Mathematics and science departments receive top priority in spite of the wealth of computer-assisted instruction for all disciplines. Computers for composing--widely used outside education--are generally unavailable for English and language arts classrooms.

Among the consequent resolves on the matter is

> that NCTE urge equity of access to computers among students of varying socio-economic levels and among various departments within a school.[6]

Those who, like a speaker at a recent major computer conference, predict that the computer will bring about "a tremendous democratization of learning," would do well to add that such a result will not be achieved unless educators plan more carefully than they seem to be doing at present (Sims 6).[7]

Word processing technology can be misused, ill-used, and overrated as easily as other kinds of computer technology. Word processors can be programmed to keep count of the number of keystrokes per hour--and some businesses in the name of efficiency have done this. One "academic humanist" notes, wryly if a bit cruelly, that word processors permit the personalization of the hundreds of job rejection letters most academic departments must send out each year (McKenzie, "Typing Chores" 25-26). And among the sappiest of word processing enthusiasts are those who do not understand the sometimes shoddy opportunism of the computer industry; because products can so quickly become outdated, some are rushed into the market before all their "bugs" are

discovered and corrected, with producers relying on disgruntled customers to do their final "testing" for them. Rather, these enthusiasts insist on seeing the industry as a resurgent instance of capitalism in its purest, most effective form:

> Intense competition for software sales dollars has benefitted the user in several significant ways. First, totally inadequate or obsolete products are eventually forced off the market. Second, good programs can be expected to sell enough copies so the amount of development cost that must be recovered per unit can be held to a minimum. Third, the profit potential offered by successful products attracts the best programmers. Finally, prices must be structured to keep programs with similar capabilities attractive to potential buyers.
>
> (Schmeltz 38)

If the seemingly miraculous operation of the word processor can sometimes induce a certain mindlessness about its uses, it should also be said that some of its intrinsic attributes might also lead to more positive results. For instance, as Zinsser says, "with the advent of computers it will be crucial for managers and other people who work in offices to have 'keyboard skills'," (27), since it will be inefficient to have to wait for a secretary or computer operator to enter each small piece of data in the office or to be unable to continue to work with computer-related projects on a personal computer at home. Such a coming together of tasks may help break down the often unproductive and undemocratic hierarchy between clerical and management staffs. In education, for instance, it might be the poorer students--precisely those who have major problems in trying to produce a clear, legible text with which to work--who can profit most from writing (and doing other work) on a personal computer. While it may be difficult to insure that, at all educational levels, such students get computer equity, if not priority, doing so is not an impossible task, given sufficient recognition of the need and adequate long-term planning.

The congressional sponsors of the recent unpassed "Apple Bill"--which proposes to fund large-scale computer training in education--argue that our educational institutions must be able to prepare students for jobs in a marketplace that will have thirty million computer-related positions by 1990. Clearly, only some massive and widespread educational effort will allow students of all backgrounds to compete for such jobs on anything approaching an equal footing. About the specific computer topic of this monograph, it is quite rightly observed that "the pedagogical applications of text editing [i.e., word processing] programs are particularly important in a world where students will do much of their professional writing on word processors and microcomputers" (Standiford et al. 23). As we shall shortly see, the benefits of using word processing in the classroom are more extensive than merely providing a very necessary preparation for the technology of the workplace.

Notes to Chapter 1

1. William Zinsser's Writing with a Word Processor is by far the most rational and lucid appraisal of the considerable advantages and occasional disadvantages of using a microcomputer with a word processing program to write. One should note, however, that Zinsser's admirable clarity and straightforwardness goes a fair way toward obscuring certain important issues he touches on but does not analyze. For instance, his last chapter glowingly describes the process whereby his book was to be quickly and efficiently printed--namely, the disks with his (already copy-edited) text would be typographically coded and page-formatted by computer, the results transferred to page-by-page film negatives, and offset plates made from these. What Zinsser points out is the considerable saving of time in this procedure (one week as opposed to three months under the older system). What he fails to point out is the considerable expense saved by the publisher, who now can employ fewer persons, since authors will be doing their own copy-editing. The question of additional compensation for authors who perform this tedious, time-consuming task is likely to become a controversial one over the next few years.

2. For example, even if work on the microcomputer does not figure directly into a course, the instructor will still need to know something about the machine's great capabilities for revision, since he or she will have to construct revision strategies and revision assignments in light of the fact that a good many students will be doing their writing assignments on a microcomputer. Similarly, in business and technical writing courses, teachers will need to incorporate some instruction in word processing, since students will have to know about the capabilities and limitations of these machines, for they will undoubtedly have to work with them, either directly or indirectly, in the business or technical positions for which they are preparing. On Sunday, June 3, 1984 The New York Times carried a seventy-six page advertising supplement entitled "A Guide to the Personal Computer." It turned out to be devoted exclusively to the use of the microcomputer in business--we may take that as a sign of the times and not just the Times.

3. Because the locution "microcomputers using word processing programs" is too clumsy to employ the several hundred times it would have to be used in this monograph, I have alternately substituted the terms "word processor," "program," "computer," "microcomputer," "personal computer," and even "machine." Unless otherwise specified, all of these can be taken as the equivalent of "microcomputers using word processing programs."

Strictly speaking, "word processors"--often called "dedicated word processors" or "stand-alone word processors"--are computers which perform only word processing. Produced by companies like Wang, Lanier and IBM, these machines cost two to six times as much as various microcomputers and are normally (but not necessarily) more powerful, easier to learn, and simpler to operate than microcomputer word processing programs. However, their cost and limitation to a single computing activity make them impractical for most classroom and personal uses. (Anyone interested in dedicated word processors might start by looking at Arthur Naiman's Word Processing Buyer's Guide, which rates and compares six dedicated word processors with eight microcomputer word processing programs.)

4. As Ayer makes amply clear, von Neumann and all subsequent computer designers and builders drew heavily on the theoretical work and machine designs of the English mathematician, Alan Mathison Turing. Two technological breakthroughs--the development of the transistor (to replace hundreds of vacuum tubes) in the 1960s and that of the silicon chip (to replace hundreds of transistors) in the 1970s--were the crucial events in the evolution of the microcomputer. Some sense of the nature of this evolution can be gauged from the fact that even a small, portable present-day microcomputer is significantly more powerful than the first computer, which filled several large rooms at the University of Pennsylvania.

5. For example, "Academic improvement, increased learner motivation, and enhanced self-esteem can occur with computer technology, however, when program directors and learning center managers train their staff to use computers humanistically" (Christ 27). The first volume in the Modern Language Association's new pamphlet series on "Technology and the Humanities"--A Grin on the Interface: Word Processing for the Academic Humanist, edited by Alan T. McKenzie--is a virtual paean to the computer, indicating that, in sum, "academic humanists" can easily and unproblematically make the machine do their benign, seemingly altruistic bidding, secure in the knowledge that this technology poses no dehumanizing threat: e.g., "I simply want to persuade the timid that the computer increases, rather than decreases, the human element in the written work" (Heilbrun 8). Heilbrun's proselytizing tone, incidentally, is hardly atypical.

6. This text is taken from "Resolution 1" ("On Computers in English and Language Arts"), which was presented and passed at the Annual Business Session of the organization's national convention on November 20, 1983.

7. The speaker was Professor Joseph Raben, formerly of Queens College.

CHAPTER 2

WORD PROCESSING IN WRITING COURSES

2.1 Word Processing, Microcomputers, and Pedagogical Change

The next few years will be the period in which educators will need to work on assessing the precise capabilities and limitations of word processors and microcomputers as pedagogical instruments. In each subject area, educators will have to create a detailed taxonomy of tasks which microcomputers might help students perform more efficiently and more accurately--and then analyze carefully just which of these tasks microcomputers do and do not lend themselves to. In those cases where the computer allows for much faster work, educators should ask if it also encourages a more extensive grasp of the subject: i.e., can the microcomputer provide ways of putting the time it saves to fruitful educational use? They should also calculate what not having access to a computer will mean in an educational system dominated, at least in its upper and middle reaches, by computer-related instruction.[1]

One likely result of this electronic educational revolution will be a reconceptualization of the teaching function. Many teachers will find themselves doing much more one-to-one teaching, leaving the rote teaching to the machines, which usually do this sort of thing better and can structure instruction to fit individual questions and needs. Thus, there will be a coming together of teaching and tutoring functions, with both teachers and students having to learn the most effective points at which to consult about the students' ongoing work.

Teachers need not, under these changed circumstances, worry about being displaced by machines; as one expert, who has worked with computer-based education for a decade, puts it, "Teacher behavior still may be the most important factor in computer-based learning by students" (Oates 32). Concerning the use of word processing and computer-assisted instruction in the teaching of writing, another educator aptly summarizes why it is that computer technology can help accomplish only so much in the writing classroom:

> Current research in the composing process shows that rewriting is of major importance, and computers eliminate drudgery of this activity. Similarly, free writing, topic-generating, and some forms of proofreading are better left to the microchips. Everyone prefers variety, however, and thus students in the classroom of the future will not sit continually before a video display, but will shuttle back and forth from teacher to computer. More and more time, however, will be spent with the latter because computer programs are getting better at teaching. (Jobst 90)

Withal, teachers will always have at least these major tasks to perform in even the most completely computer-based writing course: 1) answering spot questions about technical, organizational, and substantive points in their

students' essays; 2) providing alternate explanations and illustrations of points when students do not grasp those provided by the computer; 3) asking leading questions and offering developmental suggestions that will help students become aware of what they need to improve as they go from draft to draft; 4) structuring and overseeing collaborative exercises like peer-critiquing; 5) organizing and orchestrating class discussions; 6) creating, monitoring, and altering as necessary a syllabus that makes optimum, individualized use of computer resources; and 7) serving as an intermediate and ultimate audience for the student writer (including the judgmental act of grading).

As Peter Elbow describes it in his valuable book, Writing Without Teachers, "Editing means figuring out what you really mean to say, getting it clear in your head, getting it unified, getting it into an organized structure, and then getting it into the best words and throwing away the rest" (38). This description of editing far exceeds any description of the editing function that can be found in a microcomputer manual or book. Precisely because writing teachers can intervene pointedly and effectively to help students learn editing in the complex way Elbow characterizes it, there will be a continued need for such teachers. Even if, as Helen J. Schwartz observes, "computerized text analysis can sometimes see surprising and problematic configurations in writing that a human master can miss (even when the master knows what she or he is supposed to do)," it is nonetheless true that "writers [must] proceed after text analysis to the human hammering out of meaning" ("Monsters" 142). For the forseeable future, writing will continue ultimately to be a communication between a human writer and a human reader.

To utilize computers in any classroom, one must learn something about them and also something about how to instruct students in their particular use for the subject being taught. What the authors of Computers in the English Classroom observe about teaching computer use in English courses will hold true for teaching computer use in almost any noncomputer subject:

> Computers are used in the classroom in three different ways: as the focus of instruction (i.e., teaching about computers), as the delivery medium of instruction (i.e., teaching with computers), and as the manager of instruction. Although it may seem that the English teacher is not concerned with "teaching about," some of the classroom applications will require a basic familiarity with that function....[Concerning] "teaching with" a computer, we should keep in mind that computers can be used to deliver the primary instruction or can be used to support instruction by providing either remediation or enrichment. Whether the computer provides delivery or support, the teacher retains the role of instructional decision-maker and, as such, must design ways to integrate the computer into the conventional curriculum. (Standiford et al. 15)

Without careful planning and integration of computer work in a syllabus, teachers of noncomputer subjects may find that their secondary task of providing computer instruction can effectively become their primary one, thereby obscuring presentation of their true subject matter. If the

microcomputer contributes significantly to education, it will not necessarily make the teacher's job any easier in the process.

2.2. LaGuardia's Microcomputer/Writing Program: A Summary Description

While I have talked and consulted with several dozen educators who are using, or who plan to use, microcomputer word processing in their writing programs in some way, the most significant part of what follows in the rest of this chapter stems from my day-to-day experiences, research, and experimentation as a teacher in LaGuardia College's microcomputer/writing program.

In 1982, the English Department received a New York State Vocational Education "implementation grant" to teach writing to certified learning disabled students in vocational education majors, using word processing software on microcomputer facilities provided the college by a much larger grant from the same source.[2] "Keyboarding" has long been recognized as a "bypass" strategy for dyslexia, and we felt that working on a microcomputer could provide this and probably other kinds of help for learning disabled students taking our highest level developmental writing and freshmen composition courses. We devised a program that combined classroom instruction with work done in the microcomputer lab.

All the sections we have taught have also contained, as a control group, several students without learning disabilities. Many of these are students who have repeatedly failed the "regular" version of a course, since we have wanted to get some comparative sense of how the word processor helps with major writing problems not caused by learning disabilities. Some in our control group are "ordinary" basic writing students taking the course for the first time in any format, because we have also felt it important to get an idea of how the microcomputer might aid general basic writers and whether they differed markedly from learning disabled students and multiple repeaters.

Our program has been generously funded. Besides released time for instructors, there are also similar funds for two learning disabilities specialists, who give all students in the program an extensive battery of tests (according to New York State-defined criteria) to diagnose learning disabilities and who confer regularly with instructors about their students and with students about the results of their tests. We also have a technical liaison from the Data Processing Department, who advises on hardware and software use and possible additional purchases, and a project director, who oversees administrative functions, organizes the regular meetings of the project staff, and occasionally teaches a program course. In addition, grant funds allow us to limit class size to twelve students, which experimentation during our first year showed was the maximum number of students a single instructor could handle in a seventy-minute lab, if he or she expected to spend time working with each of the students individually. In sum, this program has provided the kind of carefully designed and monitored, small-group learning situation that learning disabled students usually need to reach their learning potential, but which they rarely find for the obvious reason that it is expensive to create. Needless to say, such advantages also help improve the performance of developmental students--and, indeed, of any students--as well.

Because of these special advantages, it is difficult for us to say exactly how much using the word processor helped students in our program to improve their writing. Nor can we simply cite comparative pass rates and grades, since our students had a very different experience from students in regular sections of such courses. However, it seems that word processing technology is the crucial aid that has unified all our efforts and given students, both learning disabled students and ordinary basic writers, a system that for the first time allows them to learn and implement an effective writing-revising-rewriting method. As we have noted elsewhere, our "overall curriculum approach was inductive, experiential, and collaborative" (Arkin et al. 25). The use of the word processor for all writing from first to final draft is just what has allowed us to get the most out of proven strategies for basic writing (and basic composition) students. Our students have not only passed their writing courses at higher rates and with higher average grades than their peers in regular sections of such courses, but they have also evidenced, in many cases, a much improved attitude toward writing, a notably more sophisticated understanding of the writing task and even, in some cases, a marked improvement in all their course work.[5]

Our results thus far would seem to demonstrate that, in general, almost all basic writing students can benefit significantly from taking a writing course that integrates sound writing pedagogy with a careful program of writing and revising on a word processor. To the extent that learning disabled students typically exhibit more instances of the kind of errors that all basic writing students share (like misspellings, omitted words, and extremely poor proofreading skills) and that they have more trouble spotting and learning to correct such errors, then working on the microcomputer could be said to help learning disabled students "more." The computer will help them catch and correct a greater number of such errors. Our results to date have demonstrated our initial assumption to be generally correct as regards the use of word processing with learning disabled students--and they have highlighted a number of other benefits, and a few drawbacks, in using a word processor as the chief composing instrument in all basic writing courses.

2.3 Word Processing for Basic Writers

Because it is not interactive and not a direct training program, word processing is perhaps easier to assess as an educational aid than other types of computer-assisted instruction. In a recent important study of the writing process, the researchers concluded that:

> Good writers respond to all aspects of the rhetorical problem. As they compare, they build a unique representation not only of their audience and assignment, but also of their goals involving the audience, their own persona, and the text. By contrast,...the poor writers were concerned primarily with the features and conventions of a written text, such as number of pages or magazine format...our expert writer simply...[spent] more time than the novice in thinking about and commenting on the rhetorical problem,[6] as opposed to spending that time generating text. (Flower and Hayes 29)

Because it automatically handles potentially distracting matters of format, layout, page length, et al., and because it is simpler to generate neat, legible text using it, word processing can free basic writers to concentrate on analyzing those aspects of the writing process that these researchers note "poor writers" ordinarily ignore. However, it is the context of computer use in the writing classroom that determines just how effective writing on a word processor can be. For example, one of our most successful basic writing assignments at LaGuardia consists of pairing students to interview and write essays about each other. After each stage--from completing a questionnaire (by hand) to finishing a penultimate draft, students exchange papers with their partners in order to correct factual errors, fill in omissions, and refine phrasing and organization. Moreover, the assignment helps create a mutual sense of audience: the novelty of, for once, having their "subject" speak back to them gives writers a good sense of the affective nature of their characterization of another person; and being a very interested reader fosters a strong sense of how oneself and others can and do react to written communication. In all this, the word processor plays only a supporting role, quickly supplying multiple copies of draft for simultaneous reading by both parties, enabling insertions and corrections to be permanently entered as the need for them is discovered, and generally facilitating the evolution of a series of progressively more accurate and more complex characterizations.

Generally speaking, the more advanced the writing course, the less extensive the benefits of using word processing in it, the exceptions being business writing and some technical writing courses, which naturally profit from word processing's primary development as an instrument of business. As writers become more sophisticated, they learn to put aside concerns with format until later in the writing process and to see outlining, prewriting, and even thinking about their writing task as means of generating more significant, finished text and saving time in the long run. They also learn to take on larger, more complex kinds of writing tasks, and using a word processor will in no direct way aid them in abstracting and generalizing, arguing from example, and integrating research and other procedures that must be learned as the writing assignments become longer and more difficult.

One writing teacher asserts, "I do not see how a composition course can fail to confront students with the question of how they know what they want to say, i.e., with epistemology, if only to the extent that it makes them aware that language is at once a tool and a trap" (Gage 621-22). His point is well-taken, provided we add the qualification that such epistemological "confrontations" become much more complex as the writing task itself becomes more difficult and far-ranging. For basic writers the word processor presents a very elemental confrontation: for perhaps the first time, students face a legible, definite copy of what they have written and must learn to stand by what they have asserted or revise it. The word processor does not impinge on the epistemological confrontations of more advanced writers to anywhere near the same extent--although, as we shall see in the fifth chapter, writing on a computer does alter the "epistemology of writing" for all writers.

The simple capabilities of most word processors and the more sophisticated capabilities of text analysis programs like Writer's Workbench (which, as yet, can only be run on a mainframe computer) can be employed to draw students' attention to any number of typical basic writing problems. For

instance, consider what Mina Shaughnessy discovered about the vocabulary of basic writers:

> Three features of vocabulary stand out in papers of the basic group: a preponderance of vague nouns and ambiguous pronouns, a dependence on basic verbs, and an absence of modification. For reasons that are not always clear, or the same, writers at this basic level often fail to name the object or person or idea they are writing about. (199)

With even a simple word processing program, a student could learn to use the "search" function to locate instances of these first two kinds of vocabulary weaknesses and, even by putting his or her topic into one or two words, have the program search to see if the topic is ever directly mentioned and if so, how often. A text analysis program locates all these weaknesses, suggests suitable revision strategies, and even offers likely substitutions in many instances.

Like a few other computer skills, word processing itself is going to become a basic skill for students of the near future. As a long-term teacher of mathematics and science education recently observed:

> If through computer technology we are able to further democratize education in the culture of the 21st century, then it is clear that the basic skills of computer use, of computer concepts, and of computer thought, must be incorporated in meaningful ways in the educational programs of the present. Otherwise, present students who hope to live and work happily and productively in that future culture will be denied access to the vast quantity of knowledge that will be accessible only to the computer-capable. (Vogeli 5)

To impart the basic skill of writing without also imparting the skill of word processing may soon became a contradiction in terms.

2.4 Word Processing for Learning Disabled Students

Because many learning disabilities--such as dyslexia, very poor auditory memory, hyperactivity--are manifested as perceptual problems, the word processor is a mechanism well-suited for learning disabled students. Keyboarding and using a video display screen demand a different set of eye-hand coordination movements than handwriting--movements many with learning disabilities find easier to perform correctly. With a neat, legible word processing printout to work on and revise, many learning disabled students can circumvent the problem of unreadable and hence uncorrectable handwriting. Having a clear draft of what they have written before them on the screen and always momentarily available in printed form makes the act of visual perception easier and less ambiguous. Hyperactive students, or those with other significant emotional interference problems, typically find the computer a more compelling focus for learning and writing, not in the least because it resembles devices like television and video games with which they are intimately familiar and which they associate with a sense of play and

relaxation. Since students in our program divide time between a regular classroom and the microcomputer lab, it is enlightening to see how much quieter and better-focussed the most hyperactive of them become when they sit down before a video display terminal.

From my observations, working on a word processor is not a panacea for learning disabilities, which are by their very nature chronic conditions. Rather the word processor provides learning disabled students with a substantial physical aid that makes the slow and repetitious procedure of revising and reviewing a performable task. A learning disabled student may need to write two or three times as many drafts as other students to achieve roughly the same result--the word processor makes doing them a practical possibility. Any teacher or educator looking to use word processors with a group certain to benefit from them could probably do no better than to choose learning disabled students.

2.5 Word Processing and Collaborative Learning

Collaborative learning, wherein students serve in a variety of structured ways as the audience for each other's work and in which they systematically collaborate with each other in completing assigned learning tasks, has been recognized over the last decade as a valuable methodology for the teaching of writing. Perhaps the leading proponent, Kenneth A. Bruffee of Brooklyn College, CUNY, notes that "formal control in writing is...an act of social adaptation" (xvii), meaning that the most useful and realistic way to learn to write is by doing so in groups where one's writing (from outline to final draft) is regularly and rigorously scrutinized by one's fellow writers, who help the writer progressively "adapt" the writing to convey his or her intended meaning to an audience. At first glance, word processing and microcomputer technology, which apparently segregates writers to work alone at their individual terminals with only the predetermined pseudo-consciousness of a program as an audience, would seem to obviate any sort of collaborative approach to learning, but if our experience in the microcomputer/writing program at LaGuardia is an accurate index, such is by no means necessarily the case.

Bruffee describes four major types of collaborative tasks: 1) paired interviews, 2) reading aloud, 3) collaborative group work, and 4) peer criticism (103-125). Although we only adopted a scaled-down and modified version of Bruffee's methodology for our program, we did make extensive use of all but the second kind of tasks. We had expected, though, that the collaborative approach would occur in classroom sessions, with the lab writing sessions reserved for students to apply what they had learned collaboratively to revisions of their work on an individual basis. For example, students might carry into the lab a detailed "peer-critique" sheet on which another student had made comments on a paper's thesis statement, paragraph organization, conclusion, mechanics and grammar, and overall strengths and weaknesses--and work on revising the paper on the terminal with this critique as a guide. The students, however, instinctively carried the collaborative atmosphere from the classroom into the microcomputer lab. Students obviously felt they were part of a group experience and helped each other learn and remember commands for operating the word processor. Often they would ask advice of each other (and of the teacher) on the clarity, phrasing, and sense of what they were entering on the machine. Having their words on a screen

seemed to foster an impression of writing as a public act, since their writing was visible to anyone at a glance. And being able to print out copies of their work in a minute or two encouraged the sharing of writing in a more complete way and allowed students the option of sharing it at almost any time they wanted.

The word processor, then, may prove to be the vital link that completes the collaborative learning circle. While it is an instrument for individual writing, it is hardly one for isolated writing. Its visual "overdetermination," its speed in revising and in producing multiple copies, and the fact that it is a powerful tool whose use students learn together make the word processor an instrument that lends itself, both psychologically and materially, to the valuable practice of collaborative learning--and not just in the writing class, but in any class where students are required to write.

2.6 Teaching the Word Processor as a Writing Tool

Because of the present popularity of computers, it is very easy to assemble a class of writing students willing and more than eager to learn how to use the word processor. With adequate planning and preparation, the teacher of such a class can bring many of the students to the point where they are willing and eager to perform the act of writing per se. What is needed to start is a thoroughgoing plan to teach students how to--and only enough to--use the word processor as a writing tool.

As one educator who has worked extensively with the computer notes, "Learning to use the basic commands of a text editor takes from a few hours to a few days, but then it becomes a routine skill" (Daiute 135). However, "a few days" might be the equivalent of four or five weeks for students in a writing class that meets in a computer lab only, say, two hours a week--and this is simply too long a time for students to devote to learning how to operate the word processor before they can use it routinely to get on with the essential act of writing. Fortunately, the standard 300 to 1000 word essay required in most composition courses is among the simplest of documents to produce on a word processor, since it usually involves no variations in format or layout: students can open a file and just start typing in material. To get students writing as quickly as possible, it is crucial to calculate and teach the absolute minimal number of commands students will need to perform the basic operations of text entry and text editing on the word processor. Paring computer instructions to the bare minimum will allow most students to work comfortably on the word processor after two hours and very expertly, with their limited number of commands, after four hours. Additional commands (like those for block moves, doublestriking, and search/replace) can be taught, on either an individual or a class basis, as the need arises.

It is decidedly a good idea not to give students a copy of the word processing program manual at the outset of the course, if ever, since such a publication is at best overwhelming, full of complex directions for a wealth of functions students will rarely need, and at worst wholly unusable, full of wordy, jargon-laden, obfuscating prose. Instead, the instructor should prepare a clear, explicit handout that will take students through the creation of a document step by step and provide a brief summary of essential commands which students can continue to use for reference. (Appendix A contains a model handout.) Teachers may also want to include some keyboarding

instruction in their course--programs like Microsoft's Typing Tutor II give this--for students unfamiliar with keyboard layout and usage, but they may find, as we did at LaGuardia, that it is not absolutely necessary: when writing a short essay that must only be typed once in anything close to its entirety, even the slowest one-finger typist does not take very much longer overall to turn out all the successive versions of an essay than does the expert touch typist.

Most students are predisposed to find the word processor, in that rather overinvoked computer phrase, "user friendly." Yet, their first few hours with it can at points be potentially offputting and frustrating. Thus, instructors in our program find they must define the limited computer task being learned and reassure students that they will soon be comfortable operating the computer--and that they will not be graded on how quickly or expertly they operate the machine. Most instructors not only repeat these points frequently in the early class sessions, but they also state them at the top of the syllabus distributed to students.

According to one estimate (Reinecke 117), ten million personal computers had been sold worldwide by the end of 1983--not a very great number when we consider their likely sale over the next decade. At present, the personal computer--at least the more powerful variety used in educational situations--is still only a half-familiar instrument for most students: they have heard much about it, probably know a few people who own one, and consider it with a mixture of interest, awe, and trepidation. To allay--or circumvent--possible fears, teachers must make initial tasks as simple as possible and provide guidance and practice in learning them. Instructional material and pedagogical approaches for computer instruction need to be based on both general cultural perceptions about the computer and the specific cultural attitudes of one's students.

2.7 Word Processing in the Writing Classroom: Success Factors

The great advantage of the word processor--as has been specified several times--is that it makes editing and producing a clean revised copy of a text many times easier and quicker than does any other manual or mechanical system. As one major study indicates, these advantages are directly translated into the improvement and sophistication of student writing in a number of ways:

> At Teacher's College, Columbia University, Collete Daiute has spent several years researching the effects of word processors on student writing. She found these programs made a number of contributions. First, she found writers were freer "to spill out" ideas directly from inner speech, since they needn't worry until later about spelling, tense agreement, inappropriate wording, or other problems; these could be mended later without having to copy the entire text. Thus, she found, thoughts could more easily be captured in the writer's natural voice. In addition, projects which were heavily formatted such as lab reports (or grant proposals) could be entered first as an outline to insure correct organization, and then "fleshed out" by inserting supporting detail.

> She also found, as have many researchers, that because a word processor makes revision much easier and faster, students do more of it. An interesting sidelight she reports is that students who used a word processor made twice as many deletions and eight times as many insertions on subsequent paper and pen writing assignments. (Wresch 797-98)[4]

Mimi Schwartz puts the matter of how computers can help student writers more generally and succinctly: "Given what current research tells us about composing, particularly revision, this machine seems both psychologically and technologically suited to help the writer write more and risk more and achieve more fully-developed writing" (27). This help, however, is not restricted to the revision stage--or rather, it consists of orienting the writer to the revision stage throughout the writing process, thereby making the writing genuinely accretive, with the best of it preserved and polished and the rest discarded.

From my own teaching experience at LaGuardia College, I see five major ways the use of a word processor intervenes positively in the writing process. Instructors working with more sophisticated writers may find each of these may matter more or less, but they will probably all be present to some extent when dealing with students who are still learning to write and who, quite understandably, still have some misconceptions, doubts, and even fears about the whole act of writing:

1) Learning to operate the word processor efficiently gives students an initial sense of accomplishment that carries over into their writing efforts. Properly scaled down, the task of learning the basic operations of the word processor can be completed within a few hours. For students with little computer experience and especially for students who have had little success in writing (or other) classes, the rapid mastery of what at first appears a complex and difficult mechanism provides positive reinforcement and "launches" them into their writing assignments almost before they realize they are writing. Many of our students at LaGuardia wrote more and wrote more quickly and easily on their first essay than they ever had on a writing assignment before. Having mastered the machine, they were less wary of trying to master the task of writing.

Seeing their work in a variety of formats--on the screen, in printouts for in-class revisions, in printouts for at-home reworking--makes students more conscious of what they are writing. The technology of the word processor is a more visual technology than that of the typewriter; it particularly encourages visual attention, since errors can be corrected very simply as soon as they are spotted. Using the "destructive backspace" key found on most keyboards allows the writer to keep the text free of typos, wrong words, and other small errors--and the instant removal of such small errors vouches for the larger corrections and changes that can be made later. For students who have never learned to type, the word processor provides the added benefit of learning the keyboard as they learn the word processor--and, as one observer notes, learning it more efficiently besides: "Microcomputers teach you to use the keyboard far more easily than typewriters do. The reason, I think, is that the cost of mistakes on a typewriter is high: one must expend

considerable energy to fix them, one becomes afraid of making them, and one slows down (Hubbard 130).

2) Having a "clean" text on the screen and always immediately available in printed form helps foster the impression that a piece of writing is a "product" that is in "process." Although the notions of "product" and "process" are usually opposed, a written text, up to the point of its completion (and even beyond, if we consider later, possible or real, editions), is really a series of products (drafts) which form a process of replacing, and presumably improving on, each other. For beginning writers especially, the look of a word-processed printout, which so much more clearly resembles something in standard printed format than anything they may have previously written, is no small part of the attraction of writing on the computer. Students can look at each draft and know that it is a completed step, that it is a communicative product that others will have to consider; despite the fact that such consideration may lead to improving that product, it is still a substantial, achieved thing in its own right. In other words, the ordered, more readable physical appearance of the word-processed text helps draw attention to what is already in the text as well as to what is lacking; it highlights the good parts along with the problems. When changes are then made, as one researcher found, they tend to be more definitive and geared to the text as a whole:

> When writers make corrections by hand, they have trouble considering them as an integral part of the piece. With the text editor, writers can have clean copies of evolving texts at any time. Thus, they can carefully consider the new piece as a whole and evaluate the overall effects of each change. (Daiute 137)

It is possible for students to preserve all drafts of an essay on the computer--to keep a record, so to speak, of the "product in process." To be able to see, within a few moments, half-a-dozen versions of an evolving essay is a great asset of this technology, and one which can be used to reinforce the students' sense that they have done much, and hopefully valuable, work in creating each "product" in the series of drafts they are "processing."

For writers, students or otherwise, who have difficulty writing in drafts--but who instead tend to eke out an essay sentence by sentence, with so much tortuous rewriting and retyping that when they are finally done they have a version that is 95% "final draft"--the computer is a real incentive to work less compulsively and more efficiently. (I speak from experience in this regard.) Awkward, hazy and otherwise unsatisfactory sentences, or even gaps, can be left in the piece, so one can get on with that better part of the writing one is ready to do at the moment; rewordings and additions can be easily slipped in later. Completing and printing out the first version of an essay, even if it will be altered by half, is a more productive day's work than struggling over two paragraphs until they are absolutely "set."

3) Using word processing as part of a formal class structure--rather than just making the equipment available for students to work on their own--allows for the immediate implementation of things learned in the classroom. Revisions suggested by collaborative tasks like peer-critiquing can be immediately, and hence more thoroughly, effected when a session at the

word processor directly follows one in a classroom. The ability of a word processing program to cut into the text at any point and allow instant, permanent changes, permits students to utilize more of what they have learned in the classroom than reworking their essay many hours (or a day or two) later. If, as one partisan of word processing puts it, "most first drafts, in fact, can be cut by fifty percent without losing anything organic" (Zinsser 99), then using a technology that entirely eliminates cuts and neatly reformats remaining text is bound to be a better way of enacting whatever revision approaches and strategies the students are taught.

Other attributes of the microcomputer could be used to implement lessons on, for example, the use of vivid, physical details in writing description. Students might have the program search their text for those vague words ("good," "bad," "happy," and, worst of all, "nice") that inexperienced writers tend to overuse, and they could then replace many of them with exact, evocative words and phrases. Similarly, students might well learn some important points about organization by taking an essay--one of their own, which would be preserved in its original form in another copy, or one the instructor had put on their disks--and moving the paragraphs around, printing out a copy of each version they evolved. Discovering the permissible and impermissible limits of reordering is much more effectively learned when a student can read a neat, complete version of each reworking than when he or she simply looks at a single printed version with a series of numbers next to each paragraph to indicate their various placements.

Although students should have an open lab period in which to do further writing, revising, and exercises, they can profit most completely by using the computer directly after regular classroom work on writing. With some advanced preparation, instructors can use the word processor as a vital (and vivid) implementation device, one that improves student writing and immediately reinforces the lesson of that improvement.

4) Using a word processor can be an important heuristic aid, helping writers to make meaning by discovering just what they want to say in the process of adapting and refining their originally imperfect way of saying it. As Flower and Hayes stress:

> ...writers don't find meanings, they make them. A writer in the act of discovery is hard at work searching memory, forming concepts, and forging a new structure of ideas, while at the same time trying to juggle all the constraints imposed by his or her purpose, audiences, and language itself. (21)

What has already been written is the raw material for this complex act of discovery: the repositioning of what has been said right, the rewording of what has not, and the supplying of what is lacking (drafts only "contain" a limited number of specific lacks) are all refinements of the original text. As one composition text so well puts it, "Writing is a process of turning information into evidence" (Van Nostrand et al. 59).

A recent article by Elizabeth Ann Sommers illustrates how word processing, combined with regular student-teacher conferences, was able to help one student profitably rework an essay through nine drafts. In the first

few, he concentrated on clarifying his thesis statement and main ideas; in the next he worked on finding a satisfactory organization; then he concerned himself with sentence-level revisions; and in the final drafts he attended to punctuation, spelling, and other editorial matters. Doing the nine drafts on the word processor took the student ten days; not only would doing so many drafts have taken the student much longer by hand or typewriter, but it is very doubtful he would have had the time in his schedule to do this many. Moreover, each draft he did rework by hand or typewriter would probably have been more completely, and therefore probably less effectively, redone. When a piece of writing has several major problems, the word processor provides an excellent means of "revising out" each one discretely. Writing may be a holistic activity, but rewriting is not necessarily or effectively so.

In her article Sommers makes clear that conferences with the students were an important part of the progress they made from draft to draft. Likewise, one of the writing teachers most experienced at using word processing and computer-assisted instruction, Helen Schwartz, emphasizes that "the protean malleability of text with word processing is ideally suited for tutoring students in writing improvement" ("Computer Aids" 239). As mentioned earlier, working with word processors in writing courses means more teaching on a one-to-one basis and less on a class basis. Students are trying, in going from draft to draft, to discover what they want to say in what they have already said, but it is the rare student who can accomplish this act alone, or even with just his or her peers for guidance. Teachers, supplemented by specially trained tutors, will more than ever have to play the role of informed audience and "writing-knowledgeable" guide as computers take over (and individualize) more and more rote instruction. Using word processing, students will be able to revise their writing more complexly and more completely--with more expert help available at key junctures.

5) Most importantly, writing on a word processor allows and encourages students to experience in full the "real" writing process--i.e., the cycle of outlining, writing, revising, rewriting--and to understand how a piece of writing is effectively shaped and completed. Students soon fall into the habit--because it is the obvious way of using the technology and because their time at the computer is divided into fixed units--of writing in drafts, of concentrating on altering different aspects of a piece as they progress from draft to draft, and of doing as many drafts as they find necessary to make the piece satisfactory to them. At LaGuardia, for instance, we found students in our program took much more responsibility for their writing than students in ordinary writing classes. When students asked for "just one more day" to work on their essay, they were not being evasive: they did go to the lab during a free period, did produce a final draft, and did hand it in the next day.

In response to the question, "What Do English Teachers Teach?" the authors of Computers in the English Classroom answer, "We help [students]...to understand writing as a process, and we patiently advise them through prewriting, writing, and revising stages" (Standiford et al. x). Much of what they subsequently say demonstrates that computers, judiciously employed, can significantly help students to understand and practice writing as a process. Certainly the physical ease of revision on a word processor is an incentive to devote more mental energy to the conceptual aspect of revision, for the revisor is not inhibited by the thought that a whole series of handwritten or typewritten revisions will necessitate the copying over of the whole piece.

The result is, as one English (Moran) teacher describes it, that students do more genuine "revising" as opposed to mere, almost pointless, "recopying."

Teaching writing as a process is difficult in itself, since going over one's writing is hard, close, repetitive intellectual work. Teaching writing as a process to students who have been through years of test-taking is more difficult work still. True-false tests obviate writing; short-answer tests truncate it; and even essay tests, which English teachers seem always to be urging on other departments, encourage a "spilling out" of as much information as possible in a short space, with only crude demands made on such matters as organization, transitions, and precision of phrasing. In such an atmosphere, word processing can help students see writing as an activity not of minutes and hours, but of days and weeks--and even of months and years. Because the technology so effectively records and produces draft after draft, students quickly learn to appreciate the value of writing as a process profitably stretched out over time, a process whose aim is not ultimately to "unload," ragtag and hurriedly, the contents of one's mind, but rather to achieve a clear, complete statement of one's ideas, a statement that can aspire to some permanence. More than any other aid to writing, word processing demonstrates the value and logical sense of approaching writing as a process; in fact, a writer has to be more than slightly obstinate not to practice, in some degree, writing as a process when using the computer.

2.8 Word Processing in the Writing Classroom: Special Techniques

Effective writing strategies, like free writing and prewriting, can just as easily be performed with the word processor as with pen and paper, perhaps even a bit more easily once students are comfortable on the machine, for the keyboard is light and quick to the touch and produces a more readable copy at the end of the exercise. A few programs have preset formats that can be employed for creating outlines; and all programs are so flexible with insertions and movement of text that they lend themselves easily to the sometimes erratic task of making a complete outline.

Steven Marcus reports notable success with "invisible writing," a simple, clever technological variation on free writing:

> Simply by adjusting the brightness knobs on their monitors, students can eliminate immediate visual feedback yet still record their ideas. The text may eventually be examined by brightening the screen display, and as usual, it is available for editing, saving, and printing... students noted that their usual pattern in composing was to interrupt the flow of thought frequently to edit and amend the language, syntax, and mechanics of their text. Invisible writing experiments suggested to them that their usual pauses obstruct their fluency and, more importantly, dilute their concentration. (156)

Like Marcus's, other special techniques derive from a recognition of the microcomputer's special capabilities (and limitations). For instance, it is productive to have students keep a writing journal on the same disk as their writing assignments, so they can record thoughts about the act of writing and

the particular writing they are doing, by making entries just before or just after doing their writing.[5]

As Robert Scholes observes, "Narration is, first of all, a kind of human behavior" (57). One of the best exercises to bring students to an awareness of this important fact--breaking into small groups and composing a group narrative, with students contributing one sentence in turn until it is completed--works particularly well using a word processor. It gives more substance to the effort by displaying results on a screen, by focussing the single writer's and several onlookers' attention on the terminal where the evolving text is being recorded, and by providing multiple copies for further group analysis and amendment. Those programs that allow "windows" to be opened--in which material from a second file can be put on the screen while text can still be entered to the first, are very useful for "writing by imitation" exercises: the text to be imitated can be put in the window and scrolled up as the student writes his or her own imitation of it.

Helen Schwartz's forthcoming textbook, Interactive Writing: Computers with Word Processing, promises to be a valuable compendium of techniques for utilizing word processing throughout the writing process. Presumably she will include a number of techniques she has already discussed briefly in writing: 1) sharing disks and textfiles among students for "peer group editing", 2) making "advice" files (by the teacher) which are readily available for students and which can be "prescribed" when a student has a particular difficulty, 3) preparing various five-minute free writing exercises for the computer, and 4) constructing on-screen tutorials to add to students' disks as necessary ("Computer Aids" 244-46).[6]

The ability of the word processor to duplicate text quickly, to project text on a screen (which as many as six students can view together), to permit text to be rearranged rapidly and in a variety of ways, and to provide a centerpoint for group activities means that this technology can absorb, modify, and often improve on teaching techniques based on the "older" technology of duplicating machines, overhead projectors, and cassette tapes.

2.9 Word Processing in the Writing Classroom: Real and Potential Problems

Even persons who benefit from a new technology also face certain problems caused by that technology. For all its considerable advantages, word processing is not without its attendant problems for students and, indeed, for all users. The most obvious disadvantage is lack of access: even if a student is rich enough to afford a personal computer, and even if that student is fortunate enough to possess equipment that is compatible with that at his or her institution, there will still be places--the library, study areas, locations off campus and away from home--where the student needs to write and must do so by hand or by typewriter. For students wholly conditioned to writing on a computer, writing any other way may prove frustrating and unproductive. "Machine dependency" is a real possibility, especially for poor writers, who sometimes see their noticeable improvement in writing as tied "magically" to their use of the computer to write. At LaGuardia, for example, a few students refused to register for the next required writing course in a sequence unless they could take it in the "microcomputer mode," meaning they sometimes had to wait a term, or more, to get on with their progress in writing--thereby dissipating much of the effect of incremental improvement.

A study undertaken at the University of Minnesota by Lillian Bridwell and Donald Ross has found that the characteristics of the word processing apparatus, like those of virtually all apparatuses, create problems for certain users:

> ...the limited size of the CRT screen (typically eighty characters across and twenty-four lines down) makes it difficult for writers to see as much of their text at one time as with paper and pen; as a result, text coherence may be hurt. Another problem is that the gadgetry available on a computer may distract some writers, causing them to break off from writing every few words to "play" with editing features. And, of course, no matter how simple the word processors are made, there will be a few writers terrified of the new medium. (Wresch 798)

There is no question that writing on a word processor is a much more cumbersome activity in terms of the equipment needed to perform it. For some, the sheer physical dimensions of the impedimenta can be intimidating, especially if the writer is doing only a short, simple piece of writing which seems out of all proportion to the equipment deployed to create it.

While many students do make more substantial revisions when they use word processors, some use the machine for what one computer-knowledgeable writing teacher terms "smokescreen revision--the tendency to think that nice appearances cover up flaws in meaning, and that facelifting changes are a substitute for changes in meaning" (Mimi Schwartz 29). The neat, well-ordered appearance of the word-processed text can beguile some students into only looking for and correcting those mistakes--letter inversions, misspellings, omitted words--which mar the appearance of the text.

Even students who have learned to revise and correct wisely and well on the word processor may have difficulty using their new-found skills if there are not sufficient open lab hours available. Good revision, even with the word processor, takes time. At LaGuardia, and doubtless at many other institutions, more classes, in a variety of disciplines, are being taught in microcomputer labs each term--meaning that some writing students find it impossible to match one of the lab's few open hours with a free period in their schedule. Although we are still able to make special arrangements for our students, it is clear that increasing use will soon necessitate some other plan for assuring open lab hours for all students.

Ideally, at any institution using microcomputers, a certain number of machines could be put in a room of their own, where they would always be available for individual student use. Failing that, at least three or four machines in the lab should always be reserved for student use, preferably set off from the other machines so as not to cause distractions for either the students working individually or those taking a class. Filling a microcomputer lab with classes that use every terminal in every period of the class day may appear economical and cost-effective to administrators, but allowing no open time for practice of the computer skills being learned seriously limits the ultimate value of the classroom instruction being provided.

Another whole problem area is health and safety. Working at a computer terminal for long stretches can cause neck strain, back strain and, especially, eye strain. (I know of several people who find working with even a high resolution screen too difficult and too tiresome visually.) In addition, as Reinecke notes, a number of computer users have developed tenosynovitis, "which causes painful inflammation of the tendons, often making those who contract it incapable of working. A major factor in repetition injuries such as these is continual high-speed use of keyboard, whether they are typewriters, teleprinters or computer terminals" (98-99). As Reinecke also makes clear, studies show that, particularly in job situations, working with word processing and other computer equipment noticeably increases stress, unless, as a number of unions have advocated, employers are willing to return some of the great amounts of time saved by allowing longer, more frequent breaks and regular rotation to noncomputer functions. Likewise, students should be encouraged to take regular breaks and not to work at the terminal for more than a few hours at a stretch.

Potentially the most hazardous aspect of microcomputer use is the low-level radiation coming from the cathode ray tube. While this radiation is lower than that viewers receive from a television screen, the computer operator sits much closer to the screen. If studies show this radiation to be a significant hazard, radiation screens (which are presently available) and other devices will have to be added to the already cumbersome equipment needed for word processing--and the user still might not be fully protected.

A final problem, and a more philosophical one to be sure, is that a majority of students take to the word processor almost too easily, too unreflectively. In my experience, "computer anxiety" or "computerphobia" is confined almost entirely to the faculty. Students can often be blase about writing on a word processor, totally unconcerned about the fact that they may be composing by radically new means and that old notions of "writing," "a text," and "to print" may have to be reshaped drastically. (Chapter 5 takes up these and similar issues.) The unquestioning acceptance of computers as writing instruments not only creates some barriers when it comes to understanding "writing" in and out of the past, but it also does not portend well for the future. There will always be some functions that these machines do less well than humans, and there will always be some people who would have these machines perform such functions anyway. Without the habit of questioning the computer's efficacy and suitability in each and every instance, society may become computerized to the point of absurdity, if not malignancy.

2.10 Other Computer Programs for Writing Courses

As more and more good programs for computer-assisted instruction in writing are created, such programs will be integrated with word processing, allowing writing teachers to "use computer programs and word processing to combine the systematic coverage of a writing class with the individualization of a writing workshop" (Helen Schwartz, "Computer Aids" 239). Many of the writing-related programs now on the market are little, if at all, more than ineffectual grammar drills, apparently written in ignorance of what any well-informed writing teacher now knows, namely, in James Kinneavy's words, "There are seventy years of research which demonstrate overwhelmingly that the isolated teaching of grammatical skills has little or no transfer to use in

writing" ("Relation" 5). (Ironically, we are reminded by the glossy, enticingly packaged computer grammar drills that "glamor" and "grammar" come from the same Indo-European root.)

Another difficulty, as Michael Southwell observes, is that "many of the poorly designed CAI materials currently available are actually tests masquerading as instruction" (166).[8] It may be, as one observer of the educational computer scene puts it, that "what computers can help teach most easily at this moment may not be what is most important for students to learn" (Lederman 1). Certainly, an experienced, successful writing teacher and an expert computer program designer have skills the other can only acquire with considerable effort, and even then probably imperfectly. The most productive approach to creating good, truly useful software for writing instruction might be for teacher and designer to work together, the teacher outlining valid writing techniques and the designer discovering ways to make them work on the computer. In any case, those responsible for designing software for educational purposes should keep in mind what Joan Baum succinctly summarizes as the attributes of effective educational programs: "They are usually complex in design, allowing for branching alternatives; truly interactive, allowing students to write meaningful responses that encourage thinking, brief in text, and graphically attractive" (5).

A discussion of all the educational software related to writing instruction would fill a monograph easily as long as this one, although at present it would probably contain a majority of negative and lukewarm reviews.[9] Still there are some good programs on the market or being developed (a number of which presently can only be run on a main frame computer) and their existence should be noted in passing, since they will, hopefully, serve as models for future development of software in the area.

Hugh Burns, while at the Air Force Academy, developed the earliest effective generative programs for prewriting. Based on his research which showed the advantages for student writers of making "a systematic inquiry using either Aristotle's 28 enthymeme topoi, Kenneth Burke's dramatistic pentad, or the tagmemic matrix of Richard Young, Alton Becker and Kenneth Pike" (Burns and Culp 9). Burns wrote TOPOI, BURKE, and TAGI, all of which prompt the student writer to develop ideas for a paper and to discover more aspects of the topic than the writer originally might have envisioned.

Helen Schwartz, of Oakland University (Michigan), has created SEEN (the whimsically named "Seeing Eye Elephant Network") for "hypothesis testing and analysis" ("Computer Aids" 241), first in response to questions from the computer, then, using the computer, in response to questions and suggestions from fellow students. Two versions of SEEN exist: one for writing papers on literary topics and one for papers in art history (developed in conjunction with Charlotte Stokes, also of Oakland University). Schwartz is now working on ORGANIZE, where students "first answer questions about their key terms: topic, thesis, audience, rhetorical purpose" (242), then proceed to a tutorial that helps them turn this material into a finished essay. According to Schwartz, "ORGANIZE, like SEEN, is a highly adaptable program so that a teacher can easily change the examples or explanations or revise the questions to suit the level and needs of his class" (244).

At Michigan Technological University, Cynthia Selfe and others have developed Wordsworth II, "a series of computer modules designed to provide supplementary, process-based writing instruction" in composition courses (Selfe and Wahlstrom 183). Future plans call for modules in technical writing, business writing, and introductory literature courses. At UCLA, Ruth Von Blum of the Psychology Department is developing WANDAH (Writers' and Authors' Helper), "which includes a word processor, a set of programs to encourage pre-writing, and a set of programs to help revising. It is her intention that pre-writing and editing programs both be constantly available to students, because she sees writing as a recursive rather than linear process; that is, students move back and forth from pre-writing to editing as they write their essays" (Wresch 796).

Other writing programs receiving favorable attention include Michael Cohen's HOMER (based on Richard Lanham's widely used text, Revising Prose); the systems (RSVP for mainframes, Camelot for micros) developed by the computer learning center at Miami-Dade Community College which issue detailed, individualized feedback "letters" for each essay a student turns in; William Wresch's Essaywriter (for the Apple II); and Valerie Arms' CREATE. There are also programs that deal with single aspects of the writing process, like the SWRL (Southwest Regional Laboratory for Educational Research and Development) sentence-combining program, Terri Paul and Dan Payne's Spellwell, and Sentence Logic by Marvin Kushner and Robert Latzer. Michael Southwell has adapted COMP-LAB, the popular basic writing workbook he co-authored with Mary Epes and Carolyn Kirkpatrick, for use on the Atari computer. Steven Marcus has devised Compupoem, whose name pretty much indicates what the program does.

Programs for text analysis form another whole branch of computer-assisted instruction for writing. The most powerful and best known of these is Bell Lab's Writer's Workbench, which is based on the precepts of Strunk and White's classic handbook, The Elements of Style. Actually, Writer's Workbench is a "collection of thirty related programs which serve as proofreader for certain common errors, give stylistic analyses, and suggest improvements where needed" (Wresch 795). It can also perform sophisticated statistical analyses of style--like listing the percentages of simple, compound, complex, and compound-complex sentences or the percentages of each part of speech in a piece. Among other uses, such data has great potential for research in composition theory and pedagogy. Grammatik is a similar, but naturally less sophisticated, kind of program specially designed for microcomputers. It will check not only for a variety of overt errors (improper capitalization, incorrect punctuation) but also for some of the simpler stylistic failings (redundant phrasing, informal usages, "overworked phrases").[10]

At the present time computer-assisted instruction has not been perfected to the same extent word processing programs have, but, as it improves and as it covers more areas and topics effectively, computer-assisted instruction may come to play an even more important role in writing courses of the future than word processing.[11] A recent nationwide survey of basic skills assessment in higher education indicates that "a full 85% of the responding institutions perceive poor academic preparation of incoming freshmen to be either very much of a problem or somewhat of a problem" and that "over 80% of the institutions which specified methods other than courses for meeting the basic skills needs of students do so through tutoring" (Lederman et al. 1, 31). Clearly some, if not most, of this very extensive tutoring can eventually be taken over (and

often made more effective) by computers, freeing both tutors and teachers for even more essential pedagogical work.

Notes to Chapter 2

1. These issues have begun to be considered in the many journals, most of them quite new, which deal with the educational use of computers: Classroom Computer News, Collegiate Microcomputing, Computers and Composition, Computers and the Humanities, The Computing Teacher, Educational Computer, Educational Technology, Electronic Education, Electronic Learning, Journal of Computers in Mathematics and Science Teaching, Microcomputers in Education, Micro-scope, Research in Word Processing Newsletter. Also useful to the educator are the popular microcomputer magazines, like BYTE, Compute, Creative Computing, InfoWorld, Personal Computing, and Personal Software, which provide regular updates and evaluations of hardware and software that might have educational applications.

2. Since the term "learning disabled students" is sometimes misunderstood--and so vaguely and inaccurately employed--it should be said that in our program at LaGuardia we define "learning disabled students" by precise technical and legal (federal and state) standards. According to Public Law 94-142, "Learning disabled students" are those

> who have a disorder in one or more of the basic psychological processes involved in understanding or in using language, spoken or written, which disorder may manifest itself in imperfect ability to listen, think, speak, read, spell or do mathematical calculations. Such disorders include such conditions as perceptual handicaps, brain injury, minimal brain dysfunction, dyslexia and developmental aphasia. Such term does not include [students', who have learning problems which are primarily the result of visual, hearing or motor handicaps, of mental retardation, of emotional disturbance, or environmental, cultural or economic disadvantage. (Brown 1)

In other words, learning disabled students have "a variety of problems in acquiring, storing, and/or retrieving information" (Brown 1). Learning disabled students have average or above average intelligence, but manifest some particular problems in listening, thinking, talking, reading, writing, spelling, or mathematical calculation--with the result that their academic performance and achievement show a marked discrepancy from their intellectual ability and potential.

Testing and certifying learning disabilities is a complex and complicated procedure, which requires an extensive battery of tests and interviews before a student can definitively be classified as learning disabled. In the LaGuardia program, we have two learning disabilities specialists who test according to New York State mandated criteria and who also interview each student. Eight measures are used:

1. a writing sample.

2. an intake interview (to judge quality and content of speech).

3. a questionnaire on background and developmental history.

4. WAIS-R (Wechsler Adult Intelligence Scales).

5. Bender-Gestalt Test.

6. Binet: Sentence Memory Test.

7. Wepman Test (for auditory discrimination).

8. Wide-Ranging Achievement Test (for spelling, mathematics, and reading/word recognition).

These instruments test students for intellectual capacity, strengths and weaknesses in auditory and visual perception, problems with memory (visual or auditory), attending and organizational abilities, and eye-hand coordination. Although each learning disabled student is unique in his or her constellation of strengths and weaknesses, in general it can be said that what qualifies a student as learning disabled is a severe deficiency in one or two of thes areas while demonstrating at least average capability in the other areas. When the capabilities of learning disabled students are graphed--as they are on the summary sheet we use in our program, their graphs typically show erratic swings, whereas those of other students tend to be more regular and level.

Learning disabled students come into our microcomputer/writing program in two ways. First, those who have already been tested and certified as learning disabled are placed directly into the appropriate writing course (almost always the top-level basic writing course, since very few of them manage to pass the course under normal circumstances). Second, students who are failing a writing course for the first time and who manifest a complex of learning disabled symptoms (e.g., auditory perceptual problems, dyslexia, poor impulse control, visual motor problems, extremely illegible handwriting) are recruited (via a recommendation sheet circulated to all writing faculty). On average, 75 percent of these students test as having a learning disability--the other 25 percent form part of the control group in each class. (The remainder of the control group is comprised of other students who randomly register for the special sections.)

3. For example, the first section of the highest level developmental writing that I taught in our program was made up of students who had failed the course an average of three times--all but one of them passed, most handsomely with A's or B's. One particularly bright but severely dyslexic student went from an F in writing and an 0.8 index to an A in writing and a 3.54 index (in which, incidentally, her A in writing played a very small part, since the course carries only half a credit). In subsequent terms our success rates were also noteworthy, if not this dramatic.

In terms of balance, it should be noted that Richard M. Collier has conducted an intensive study of the effects of using a word processor on a

very small group of student writers. He has concluded, among other things, that "the text editor [word processor] did not encourage the subjects to make serious and elaborate additions to their essays" and that "the use of a text editor was least successful for the subjects whose writing skills were the weakest" (152). It may be, however, that since Collier was dealing only with students in an introductory composition course, his judgment in the latter regard might not hold for remedial and developmental writers.

4. A study of beginning writers at Montana State University turned up essentially the same results: "The success of this program for students suggests that the computer can help beginning writers learn to revise their initial drafts with less emphasis on lexical substitution and grammatical correctness and with more emphasis on progressive reshaping of ideas through successive drafts" (Bean 146). Although it had a number of qualifications, the study by Collier did find the positive elements to be basically the same as the Columbia and Montana State studies, including the facts that:

--"The subjects made two-thirds more substitutions, engaged in twice as much reordering, and reduced their deletions by two-thirds."

--"Particularly outstanding was the increase in substitutions, especially of one word for another and of one T-unit for another."

--"The subjects engaged in more revision, considering and experimenting with a great many alternatives" (151).

5. Apple's new Macintosh computer has a feature that makes such journal keeping even easier: an "electronic notepad" on which up to eight short entries can be recorded while the user is working on a file. This permits the "jotting down" of momentary thoughts for later retrieval and inclusion into the writing journal.

6. Carol McAllister discusses several other useful techniques including a computer-assisted way of working with student drafts and a deletion exercise.

7. The opening sentences of the company handout heralding the arrival of the Macintosh demonstrates just how oblivious those who would remake the world in the computer's image can be to justifiable fears about the coming of an "Orwellian" nightmare electronic state: "In the olden days, before 1984, not very many people used computers....Then, on a particular bright day in California, some particularly bright engineers had a brilliant idea: since computers are so smart, wouldn't it make sense to teach computers about people, instead of teaching people about computers?"

8. One example of this failing is a program entitled Writing is Thinking, which purports to be a computer-assisted method for teaching beginning writers the 500-word essay. Instead, it merely tests at several points students' ability to write a predetermined number of short statements--of no more than 256 characters (a severe limitation)--about their chosen topic. Concerning writing, the instructional material is discouragingly prescriptive ("Know what you mean to say"), unduly doctrinaire ("Facts are the same for everyone"), at best very generally directive ("Allow Ideas to Flow Smoothly"), and at worst entirely unhelpful ("Correct All

Grammatical Problems" is followed by exactly one example, showing how comma placement can change meaning). The writing is also so laden with exclusive use of the male pronoun that it is difficult to see how any woman student could feel she had a part in the process being described--e.g., "When the writer is ready to structure his paper, he has five patterns from which to choose. The pattern he chooses will give him a predetermined plan for laying out his presentation." A second version of Writing Is Thinking is under development, but apparently this will address chiefly its computer limitations and not its substantive lacks.

9. There are several publications which have listings, reviews, and ratings of educational programs. For example, Educational Software reviews and rates nearly four hundred programs in twenty subject areas at all grade levels. Their ratings, though, do seem a bit generous considering what is on the market.

10. According to a very recent computer book, Grammatik "has been pulled from the market for rewriting" (Gonzales 66).

11. Anyone interested in doing more reading on the variety of ways both word processing and computer-assisted instruction are being used in teaching writing might start by looking at the annotated bibliography compiled by Thomas Mulder.

CHAPTER 3

WORD PROCESSING FOR GENERAL FACULTY USE

3.1 Faculty Resistance to Computers and Word Processing: A Mild Polemic

There are probably very few faculty members, even among those not regularly doing research and writing for publication, who do not write a good many things during any given academic year: syllabi, exercises, bibliographies, handouts, tests, directions, memos, reports, and the like. Unless they are members of those rare departments generously staffed with experienced clerical personnel, faculty often end up competing with each other for secretarial assistance--and more than occasionally end up typing, retyping, and copying their own work because the "queue" is too long and they need the work in a hurry. Learning to use the word processor would greatly help faculty to circumvent such situations, which is not to say every faculty member needs his or her own personal computer (some do), but only that almost all of them would find academic life easier and more pleasant if they at least were able to make use of the word processing facilities at their institution.

A simple personal example might serve to illustrate this point. At LaGuardia, faculty are requested to include a number of things on all course syllabi: "instructional objectives," "performance objectives," grading standards, attendance policy, etc. For writing courses, I also include a long list of correction symbols and sometimes a page reference guide, so students can quickly locate in the assigned textbook discussions of particular problems they may be having. When detailed class schedules and other material are added, the syllabus runs five or six single-spaced pages, in an irregular format that is slow to type and prompts many errors. Also, there are just enough changes in detail in the two-thirds of the matter that is repeated so that it cannot be cut and pasted for photocopying, but must be retyped completely. When I am teaching three different courses, preparing neat, clear syllabi for them can take the better part of two days if I do all the typing. If I enlist secretarial aid, I have to allow at least a week, and usually two, for the work to be completed and copied. In either case, the process involves boring, tedious, and repetitive work, which is not only deadening in itself, but sometimes has a debilitating effect on more imaginative work done after it. However, if earlier versions of each syllabi were on a computer disk, preparing the three new versions would be a matter of just a few hours. Altering details and automatically reformatting standardized matter could be done very quickly, and the schedule could be typed in from very rough notes (saving the time it takes to make a complete, legible handwritten copy from which a secretary can work). If, as often happens, the class schedule is changed during the term, creating and printing out a revised schedule would take but a few minutes, thereby eliminating the time spent and inevitable confusions caused by having students enter changes by hand on the original schedule.

The point of this example is not to tout the "wonders" of word processing but simply to illustrate one very typical way word processing can help eliminate time-consuming, unimaginative work from an academic's schedule. If

he or she has no symbolic worries about doing "menial" secretarial work, the academic can also gain much more control over work flow by using word processing, which provides a complete system for entering, altering, combining, and printing out (in multiple copies if necessary) material. I would estimate the average faculty member could save at least ten full days of work per academic year by employing word processing. For the faculty member regularly engaged in research and publishing, the figure would be three to four times as high.

Clearly, then, the evident resistance of many faculty to the computer is more symbolic than practical, and it touches deep-seated and long-standing cultural notions, particularly for faculty in the humanities. The character of these "anti-mechanistic" attitudes is, in the specific sense of the term, "Romantic," a product of that intellectual (and anti-intellectual) movement roughly contemporaneous with the Industrial Revolution and in many ways a reaction to it. As Raymond Williams delineates it in his clear-eyed study, Culture and Society: 1780-1950, Romanticism created a series of dichotomies that still condition our thinking: "The contrast between 'grows' and 'made' was to become the contrast between 'organic' and 'mechanical' which lies at the very center of a tradition which has continued to our own day" (37). As a result, there is an ever present fear of encouraging a situation in which, to invoke Emerson's words of a century and a half ago, "Things are in the saddle/And ride mankind." Despite professions of other intellectual faiths--Marxism, historical relativism, structuralism, Freudianism, semiotics, deconstruction--many academics still preserve at the same time a faith in the reality and moral significance of this "organic-mechanical" division. In such an intellectual climate, an electronically created, "word processed," automatically and mechanically printed text is bound to appear the epitome of the mechanical intruding on and destroying the organic.

Williams also makes note of the "very familiar Romantic literary theory: contrasting the spontaneous work of genius with the formal imitative work bound by a set of rules" (37). It is, of course, gratifying to think of one's academic work as being, if only lightly, touched by genius (this clever exercise, that brilliant essay), but in fact very much of academic writing, of all kinds, is "bound by a set of rules" and would be useless were it not. Getting faculty to acknowledge this fact might well move them toward a more rational appraisal of what using a computer could and could not contribute to their work. There is no particular reason a word processor, complex and intimidating though it may first appear, should not eventually be used to capture "the spontaneous overflow of powerful feelings," to cite Wordsworth's influential characterization of poetry. It is also handy for less grandiose tasks.

Still, faculty who resist the computer in general and word processing in particular have a point. Writing on a computer is not, as too many boosters of word processing would assure us, making use of "just a more powerful typewriter"--rather it is writing in a radically new, wholly electronic way. For anyone used to the older technologies, from ballpoint pen through the fanciest electronic typewriter (which, in essence, still produces a text manually), the changeover to a word processor can mean the breaking of a number of not always productive writing habits--to say nothing of the possibly disquieting effects it can have on the writer's whole notion of "writing." Moreover, writing on a computer in some way aligns one with those forces that,

for good or (more probably) ill, would have computers determine very many aspects of our lives--and determine them in ways too complicated for most ordinary citizens to comprehend or even recognize. Yet, in this regard we are in the realm of symbolic behavior: using or not using a word processor will have very little, if any,[1] impact in those areas where the computer is capable of doing great harm. Taking the time to learn some basic things about computer operation would be a much more effective way to assess and possibly influence the direction of the computer revolution rather than making the symbolic gesture of refusing to acknowledge it.

To resist learning anything about word processing and other basic computer functions also means doing a disservice to the coming generation of students. For instance, writing on a word processor will fairly soon be the norm, on the job if not always at home, and to teach students to write without taking this situation into account will mean highhandedly imposing one's own archaic writing habits on students.

Some, perhaps most, of what is counted as faculty resistance to computers might better be characterized as disinterest and indifference stemming from misconceptions and lack of knowledge. "Word Processors were what happened to somebody else" is how one teacher-writer typified his own, since changed, apathy about using a computer to write (Zinsser 10). Like him, most faculty will not become convinced of the limited but specific advantages of a word processor until they use one over a period of time to create, revise, and update "real" documents. For instance, grant proposals, which are often written collaboratively, piecemeal, and against a deadline that always seems a week too soon, are much more easily and much better completed on a word processor. Resumes, which ordinarily need to be retyped in their entirety every six additions or so, can be kept up to the moment on the word processor with hardly any effort. Moreover, different versions of a resume, with different emphases, can quickly be created as needed (a very handy way of highlighting one's qualifications for a particular fellowship, grant, or job). For writers who regularly work in specialized formats, there are programs which will make the task easier by automatically setting up the required layout (like Scriptor or a special version of PIE Writer for plays and screenplays). A scholar who wants to write in English and Spanish in the same manuscript can do so with Select Bilingüe.[2] There are a few word processing programs which have the capacity to create indices and tables of contents--and supplementary programs to perform these functions for you if your main program cannot. Perhaps most importantly, writer-academics will find that word processing allows them to finish sustained pieces of writing more quickly and to revise them more thoroughly and exactly--advantages particularly important for persons who most often are writing in fits and starts because they are almost always doing something else equally important, namely, teaching.

The benefits of word processing are evident only over time--a faculty member who has two or three hours of orientation and practice is not going to be convinced of its advantages. But when a faculty member realizes that he does not have to type a whole "101" syllabus for the umpteenth time, that she can effect changes an editor requested and reprint a new version of any essay in less than an hour, or that he need only replace a few details to have an updated version of a list of department procedures, then the benefits of word processing will be seen and appreciated. It is just in saving faculty hours and hours of tedious, boring recopying that word processing demonstrates its

value for the institution. And it is in allowing scholar-writers to turn out more carefully worked essays and books that it demonstrates its value beyond the institution.

3.2 Writing vs. Recording on the Word Processor

Although I have been assured otherwise by several persons in the position, I still believe that a majority of academics and writers who have practiced writing for many years using pen, paper, and typewriter will not go over completely to working on the word processor. When it comes to more complex pieces of writing, most of these writers will use the word processor more as a recording instrument than as a writing instrument--i.e., doing a majority of their first draft by longhand or typewriter, then making minor adjustments as they enter that material on the word processor, leaving major revisions for the series of printouts the equipment will so efficiently provide afterwards. Whether the insubstantiality of the electronic text, the distraction of the equipment, or simple obstinancy is the cause of this general avoidance of the machine at the "creative" stage--and whether this situation will change for many with time--I cannot judge. However, from observing my younger and less habit-bound writing students, I would guess there will always be a certain percentage of writers at all levels who will still want to do some of their writing work manually (as well as some who will insist on trying to cart their computer everywhere, since they will not be able to envision writing even a short note without it).

For faculty, or indeed anyone, worried about the word processor disrupting good personal writing habits, it can be said that a little forethought will allow a writer to integrate word processing in a writing pattern without having it dominate or destroy that pattern. Once they have become comfortable using it, most writers go from a considerable wariness about the word processor to a real dependence on it. For a while, hopefully a short while, that dependence can be troubling, since it can sometimes focus too much attention on what a piece of writing looks like and not enough on what it has to say and how it says it. Yet, any writer done permanent damage by the application of a technology to his or her writing may well be a writer not certain or sure enough to be worth reading. We should recall that what is generally credited with being the first American novel written on a typewriter is also generally credited with being one of the great expressions of the "natural" genius of the American vernacular: The Adventures of Huckleberry Finn.

3.3 Word Processing in Higher Education: Some Effects on the Professoriate

The Modern Language Association, for one, is sponsoring a view of word processing that is unequivocal about its benefits (and about who should foot the bill so "humanists" can enjoy them):

> The word processor is a powerful new tool that ought to be in the hands of every humanist, and colleges and universities in which humanists teach ought to put it there. (McKenzie, "Introduction" ix)

> Only humanists with breathtaking calligraphy, efficient secretaries and no second thoughts can afford to ignore word processing. (McKenzie, "Work Habits" 11).

Obviously, academics, and not just those of the humanist stripe, have already and regularly made some good use of computer resources, in some cases only half-aware they are doing so. For instance, their requests to the interlibrary loan office might have been processed through the Ohio College Library Consortium, which enables a book to be located rapidly in any of 3,300 libraries. Or they may have employed computer-generated statistics in a grant proposal or report. Certainly, they have worked with computer printouts, like course enrollment tallies or grade distribution rosters, produced by the registrar's office.

Such functions are, in one way or another, administrative and data-producing. Word processing on a microcomputer--or on a mainframe for that matter--provides the individual complement to these functions. The memos, reports, essays, and books which eventually result from a computer-based collection of information can be done (and updated) much more easily and quickly if the word processor picks up where large institutional and inter-institutional computers leave off.

In some ways, the academic writer who does turn to word processing when ready to write will have experiences similar to students and other writers using a word processor. What one academic writes about working with the printout of his first draft describes an experience that almost all writers using a word processor will go through, usually to their notable benefit:

> I can see it, hold it and write on it. That difference makes my reading of it a new experience and allows me to see the text in a new way; I do not feel as if I am reading my text again; rather, I feel as if I am reading a text for the first time, for it is the first time I have seen it in that physical form. As I read it, I freely mark it up, making suggestions for revision, adding notes about ideas in the margins, editing sections of the text. I am not at all hesitant to be a severe critic. I can return to the words in light and easily and swiftly manipulate them into a new shape. (Self 20)

However, to the extent that they are more experienced and polished writers than students, academics will find using a word processor for writing different in a number of respects. First, they will regularly be working on a much broader range and variety of documents, from three-sentence memos to chapters in a manuscript of many hundred pages; thus, the periods of time they spend completing a document can be a matter of moments or a matter of months and years. Second, the variety of documents they will be producing means that academics need to learn more about the many capabilities of word processing--and probably need to use more sophisticated programs than students. Third, academic writers will be writing and revising on the word processor without the steady outside feedback that student writers will, or should, be receiving; in other words, academics will be taking advantage of the technology to write and revise more or less on their own (probably sharing later drafts with colleagues, as they have usually done), whereas students

will use the technology to write more interactively, very regularly sharing work, at all stages, with peers and teachers. Fourth, academics will probably have to adjust to a somewhat different schedule when it comes to writing for publication, i.e., documents for publication will be easier to produce and will be produced more quickly, but they may take even longer to be read by publishers, since the advent of word processing is likely to increase the overall number of submissions. Fifth, some academics will experience a change in their style, brought on by the facility of the word processor in producing text. As one teacher-writer describes and, evidently, demonstrates it: "My style has become somewhat more open (some might say breezy) and the writing has more of the rhythms of the spoken language in it. This is not necessarily a loss or gain, but a fact to be noted and dealt with, as appropriate" (Moran 114). Sixth, even if they do not get involved in writing instructional programs, many teachers will probably find themselves creating exercises and assignments for use on the word processor, meaning they need to consider what might be termed the "general psychology of word processing" as well as their own individual reactions to writing and working on the equipment.

Finally, some academics might find themselves involved in using word processing in conjunction with publishing functions. Specifically, word processing may prove crucial for the hundreds of specialized, small-circulation scholarly journals published by colleges and universities. As two academics concerned with the fate of such publications describe:

> For the editor of a scholarly journal facing rising production costs and decreased institutional and personal budgets, however, the use of word processing and other computer capabilities to reduce production costs, staff size, and publication time may well be a matter of survival. (Haselkorn and Borck 49)[3]

The abilities of the more powerful programs to do formats and layouts very close to those of standard printing will mean that journal staffs can virtually become their own printers.

Those academics who do get involved in creating computer-assisted instructional materials are, as yet, treading on very uncertain academic grounds as regards their careers. As Joseph H. Bourque noted recently, and quite importantly, in College English:

> Serious morale problems could develop as computer uses accelerate if departments do not set up reasonable guidelines for evaluation of this non-traditional work. With this essay I seek to initiate discussion of just one issue relating to the use of computers in departments of English: the value of computer-related work as demonstration of professional development for tenure, promotion, or merit. A distasteful topic, no doubt, but we have laboriously set up evaluation systems just because we do cherish the continued vigor of our dedication to knowledge, in the classroom and in our own minds. The question becomes, then, to what extent computer-related work contributes to advancement of knowledge in our

> discipline and to what extent it should be encouraged and rewarded.[4] (68)

A further problem in this area is that it will take computer-knowledgeable faculty to evaluate their colleagues' computer-related work adequately, a fact which could split some departments into two factions, with the traditional faction too summarily dismissing computer work as insignificant, the computer faction too easily championing all such work as important.

One way for the computer faction to gain the respect of its opposite is to avoid the temptation to speak endlessly, esoterically, and glowingly about the computer's great capabilities.[5] Rather, the computer users should stress what they are producing rather than how they are producing it. After all, a scholarly article, classroom exercise, textbook, grant proposal, or institutional report done on a word processor is intrinsically no better or worse than one produced by any other method. Its producer may know that it is just a bit better because the technology allowed for some extra refinements and final adjustments, but those who judge it good or bad will, all things being equal, judge it absolutely.

3.4 Word Processing Programs for Faculty Use: Some Points to Consider When Choosing

One school of word processing thought has it that you should first select the exact program you need, then locate the hardware to run it, since the best machine you (or your institution) can afford may not be the best equipment to run that program; in fact, that equipment might not even be able to run the program at all. The opposing school urges that the microcomputer is by far the more expensive and important of the two purchases and so should be given first consideration, especially since it is difficult to know just what word processing program you want until you have had a good deal of microcomputer experience. A third school argues that the decision can be obviated by finding a computer that comes with "free software" that will probably meet your needs--e.g., the inexpensive Kaypro II comes with WordStar and a number of other programs, the cost of this software, were it purchased separately, almost exceeding the cost of the computer itself.

All these viewpoints have validity--and it is not possible to reconcile them completely. Each faculty member and each institution that comes to the point of buying hardware and software for word processing will carefully have to balance cost factors, real and anticipated word processing needs, probable "wear-and-tear" on the equipment, portability, compatibility, and the possible use of the microcomputer for functions other than word processing. Since cost is usually the key factor, both individually and institutionally, it might be said that faculty demands on word processing are normally more complex and diverse than student demands or even normal clerical demands, meaning, obviously, that more rather than less must be spent to meet faculty needs--a good thought for faculty to keep in mind when, or if, the institution surveys their microcomputer needs and desires.

Since it is not possible, ultimately, to choose a word processing program completely apart from hardware considerations, there are some elemental facts about microcomputers that any potential user ought to know. To start at the end, you will need a printer to turn your electronic text into a printed one,

and printers are almost always a separate item when purchasing a microcomputer. Printers are of two kinds: dot matrix and letter quality. The former are usually very fast (and very loud), forming characters by spraying ink through the appropriate holes in a tiny rectangular matrix of dots (8X6, 9X7, and so on). Letter quality printers, which operate via either a daisy wheel or thimble, strike characters on a page as does a typewriter and produce copy indistinguishable from that produced by the finest typewriter. (IBM Selectric typewriters can also be adapted to print out text from a computer file, but they do it very slowly.) On average, letter quality printers cost three times as much as dot matrix ones and run at one third the speed. They are, therefore, impractical for most personal and institutional uses--but they are very helpful when it comes to printing final drafts of manuscripts, fellowship applications, grant proposals, institutional reports, and other documents which need to look as good as possible. Many institutions have a few letter quality printers to supplement the many dot matrix printers that do the bulk of the work; any faculty member ready to buy a personal computer would be well-advised to check to see that it is compatible with a letter quality printer available at his or her institution. A less expensive alternative to a letter quality printer would be a very high quality dot matrix printer--one with at least eighty dots to the matrix--which will produce legible, distinct copy that is very close to that of a letter quality machine, save for the slightly more squarish look of the characters. Be warned though: while the resistance of editors, publishers, and those who read for them to dot matrix manuscripts is fast disappearing as dot matrix printers become better and better, there are still many places that will not accept a dot matrix manuscript.

A useful rule of thumb concerning hardware is that good graphics capabilities usually mean poor word processing ones. Fancy, expensive color monitors should be avoided if the microcomputer is going to be employed chiefly for word processing: even an inexpensive monochrome display will provide clearer, sharper resolution. Color monitors can be fetching to the eye, but an academic writer must remember that he or she will be spending thousands of hours looking at the display screen and the less eye strain caused thereby--all monitors cause some eye strain--the more useful a writing instrument the word processor will be.

Of course, the general consumer rules for major purchases--and a microcomputer can cost as much as a small car--hold full force for anyone buying a personal computer. Make the salesperson explain the hardware and software thoroughly and completely; arrange as much time to try out the computer and program as you need (or can get); check warranties, guarantees, and especially service policies and service contracts very closely; speak to people who have used the hardware and the software; and be sure you really want these products. (Might not your institution's resources be adequate, for the moment anyway?) Also, when testing hardware and software try to create an indicative variety of the kind of documents you regularly write; do not just randomly type in and print out whatever comes into your head.

When it comes to judging the power and flexibility of a word processing program, the roughest measure is its total number of commands, with the most sophisticated programs having about 200. Normally, at the back of the computer manual, there will be a comprehensive list of all commands, a list which will yield a total and give a quick notion of the things the program can

do (and even something about how it is organized to complete tasks). The manual will also provide a good deal of other information about the program's structure, capabilities, "logic," and limitations.[6] Withal, reading the program manual in no adequate way substitutes for working with the program on the computer. Extensive capabilities are of no value if they are very hard to use; a program, whatever its virtues, is almost worthless for writers who cannot get its operation to mesh easily with their thinking and writing habits. Good, clear program designs do not always translate into clear, direct prose in the manuals accompanying them: what sometimes seems obscure and unfathomable in the manual will prove simple and clear on the machine, and what seems clear and evident in the manual may prove difficult and confusing at the terminal. Moreover, much of the material in the manual will make no real sense until you have worked with the program for a number of hours; for example, it would be very difficult to judge why or for what reason you might want a program that offered programmable function ("soft") keys until you had used the program for enough hours to let you gauge what else you might want to add to its regular functions.

The author of Computers for Writers asserts that it takes five ninety-minute sessions to assess a word processing program adequately and accurately (Gonzales 90)--and in this he is correct, except that a novice will need nearly twice this long with the first few programs, after which he or she will then know just what to look for. (Make no mistake of it; learning to operate a word processor and choosing the right program to run on it call for a substantial investment of time at the beginning, an investment that will, however, pay itself off rapidly.) Bringing one's necessary work to the word processor--a test you are preparing for the following day, the draft of a (short) piece you are writing for publication, or, if daring and undaunted, your résumé--will provide not only a realistic test of the program's overall use to you, but will also allow you to see the program from a different, more personalized, angle than the general view of it presented in the manual. When testing a program, it is best to spread sessions out over time. (Two weeks would be ideal.) That way you will be able to judge not only whether program commands are easy to learn and execute, but also whether they are easy to retain and recall.

Speaking of the whole purchase of hardware and software, one word processing expert states, "From the beginning, you must isolate what you expect the system to do as a group of processes (Townsend 13). What this means for academics trying to choose the best word processing program is that they must first review very carefully all the kinds of documents they regularly produce, closely analyzing all the editing and formatting demands these documents entail. Obviously the initial list this analysis yields will have to be expanded as testers discover, in practice, more features they need or want, but listing basic demands at the outset will help eliminate some programs right away. For instance, most academics do writing that requires shifting from double-spacing to single-spacing (for long quotations) and so could exclude from consideration the many programs that require spacing to be standard throughout a document. Conversely, many academics might be pleased to have a feature that allowed for automatic on-page footnoting, but not many, except those using word processing in tandem with publishing, would absolutely require this feature.

For the uncertain, there are inexpensive, scaled down programs, like Select Write, which can later be traded in for the full-fledged version, with the original purchase price given as a discount. It is possible, too, that any academic's word processing needs will change over time or that more sophisticated and powerful word processing programs will be developed--and that an originally adequate choice will no longer obtain. By then the academics or institution will probably have gotten their money's worth out of even a $500 or $600 program. For now, it is important to evaluate some of the better programs that are available, in all price ranges, for both student and faculty use.

Notes to Chapter 3

1. Similar oversimplifications can be found in the thinking, so to speak, of computer enthusiasts, the "Dedication" to Naiman's Word Processing Buyer's Guide being a representative instance:

> This book is dedicated to everyone
> who works on uses for computers that serve people
> (like word processing)
> and not on uses that oppress them
> (like guidance systems for weapons). (iv)

2. Of course, the user has to have a printer that supports Spanish characters. Two recent articles in the MLA Newsletter (Spring and Summer 1984) discuss the several programs presently available for multilingual word processing and the many "stopgap" ways of adapting various English-language programs for foreign language use. However, as the surveys make clear, academics working in foreign languages are still at a decided disadvantage concerning word processing when compared to their colleagues who work in English.

3. The authors describe, in explicit and convincing detail, the steps by which articles can be written, revised, transmitted, proofed, formatted, and photo-offset by computer and making use of a "modem" (a device that allows the transfer of material from one computer to another over telephone lines).

4. Those who would, all too simply, equate computer work with more traditional kinds of academic work should bear in mind the still problematical status of computer work, a point which Bourque clearly illustrates; e.g., "In addition to mastery of subject matter and methods, the CAI specialist must learn one or more computer languages, a time-consuming process. Though some graduate schools have begun accepting computer languages to fulfill their language requirements, computer programming languages are not as difficult to learn as natural languages, so we should not pretend that the accomplishment is as impressive as learning French or Italian" (69-70).

5. One of the traps academic computer users can fall into is a fascination with "souping up" their computers to perform at a speed that is, in the overall scheme of their work, meaninglessly faster. This kind of thinking about computers is much encouraged by commercial publications on word

processing. For instance, in Computers for Writers, Laurence Gonzales, in a section titled "Achieving the Speed of Light," complains about having to wait sometimes "a full second" while one screen replaces another (107); in "How Slow is Slow?" he finds great utility in a method that can check a 3,000-word document for spelling errors in one minute, as opposed to the two-and-a-half minutes the normal method takes. And when he gets to comparing the increased processing speed attained by using a "RAM disk" as against using a hard disk, he admits to having "to use a stopwatch to tell the difference at first" (111). Worrying over such miniscule time savings seems a trite proposition--and an expensive one if the matter is taken seriously, since savings of seconds can mean buying hundreds of dollars of additional equipment. Chattering about such differences can rightly cause colleagues who are not computer-knowledgeable (and even some who are) to suspect one of being up to no substantial work.

6. If a manual cannot be borrowed, it can usually be bought for about 10 percent of the program purchase price, with that amount discounted if you buy the program.

CHAPTER 4

EVALUATIONS OF SPECIFIC PROGRAMS

4.1 Principles of Analysis

The evaluations that follow are purposely unlike--and so are meant to complement--those reviews of microcomputer word processing programs which appear fairly regularly in computer magazines like Personal Software and Personal Computing. Typically, these reviews cover many programs and provide a checklist to indicate which of dozens of editing and formatting features each possesses.[1] Since new versions of programs are regularly released (sometimes more than one a year),[1.1] anyone ready to purchase a program should begin by locating a few of the most recent of these review articles. Their checklists will provide a basis for deciding which programs might fulfill one's pedagogical or personal word processing requirements. To understand the terms used on these lists, the potential buyer could look at Townsend's Exploring Word Processors (30-43), which gives clear, well-organized brief explanations of all editing and formatting features, or consult the very full glossary of Naiman's Word Processing Buyer's Guide (281-310).[2]

The programs analyzed here have all been tested, at length, on the computer. Thus, analyses cover not only what features each program has--such features could be, and for reviews sometimes is, determined strictly from the program manual--but also how easy each function is to learn, perform, and remember, how well or badly functions are organized, how comfortable the program is to work with, how much it fosters concentration on the actual writing, how it meshes with various mental habits, and a number of other things only possible to judge after using a program for a number of hours. Programs are considered both for student and faculty use,[3] although points concerning the potential use by either are intermingled. In almost all cases, a description of how a program function is performed accompanies the judgment as to whether that function is easy or hard to reach, part of a logical sequence or not, and a special benefit or a program limitation.

Disagreement on how crucial a part of the word processing program plays in writing on a computer ranges from Gonzales' insistence that "there are only two word-processing programs to consider for serious writers: WordStar...and Perfect Writer" (62), to Heilbrun's laissez-faire belief that "you learn to live with the program you start with" (4). The truth, not surprisingly, lies somewhere in between: most writers, student or academic, can adjust to and make good use of a variety of programs (unless the program lacks an absolutely vital function for them), but how fast, how easily and how thoroughly they do adjust can vary widely with the program.

The following evaluations are based on many of the principles laid out, explicitly or implicitly, in the first three chapters. They are also, inevitably, subjective in some measure: when I say "this is confusing" or "that is easy to learn," I mean, first and foremost for myself.[4] However, extensive classroom experience with word processing and consultation with many faculty who have used word processing (both personally and pedagogically) also

gives me, I believe, a valid basis on which to generalize--and there is generalization aplenty in the evaluations.

A number of word processing features, which might be important in another situation, are downplayed or passed over in these evaluations. For example, "spooling" (the ability to edit one file while printing another) seems more relevant to the typing pool than the classroom or home office. Since it is assumed that users will have monochrome monitors, the color capacities of programs are largely ignored. Because removing a sentence via word and line deletions is so easy, I do not always indicate when programs also allow deletion by sentence. Similarly, if a program can delete marked blocks of any length, it does not seem very significant if it can likewise do so by page designation.

Other program features are moderately important--like the existence and organization of the status line (which indicates cursor position, the present mode, the percentage the file is "full," et al.), the extent to which commands are mnemonic (i.e., easy to remember because they resemble their result, as "P" for "print"), or the presence of "delete buffers" (which store the most recent deletion and allow for its recall if a mistake has been made). Many of these moderately important features are covered for each program, particularly if their presence, absence, or operation is in any way unusual or significant. (For example, powerful as it is, WordStar has no delete buffer.)

Since the ability to have alternate line spacing in a document (the body double-spaced, long quotations single-spaced) is a very common requirement for academic writing, the absence or presence of this feature is always noted--so too with superscripts and subscripts. When a program has "windows" (which permit two or more texts to be viewed on the screen simultaneously), this feature is always mentioned, since it can have important pedagogical applications. Also, when programs permit functions that might be needed for creating formats for publishing--most particularly on-page footnotes and double-column printing, such features are noted. Since it will inevitably prove vital, the documentation accompanying the program--manuals, reference guides, "quick start" courses, tutorials--is always assessed. Because word processing is still a very tenuous proposition for those working in the foreign languages, mathematics, and the sciences, it can be assumed that these programs have no special features for work in these areas unless such features are specifically discussed.

Despite the many, often small distinctions these evaluations make between programs, it should be remembered that these programs are in essence more alike than different, especially when working with any one of them is compared to working on a typewriter. As one enthusiast puts it, clearly if not felicitously:

> ...the Beverly Hills version and the no-frills version have this in common: They will take most of the pure mechanics out of writing whatever it is you are writing and free you up to concentrate on the real business at hand which is, after all, getting your carefully chosen words down in such a way as to do the job you wanted them to do. Period. (Keogh 170)

The evaluations that follow are divided into three categories: Inexpensive Programs (up to $125), Moderately Expensive Programs ($175-349), and Expensive Programs ($375-495). Be aware, though, that price changes are common and that producers are constantly releasing new, and supposedly, improved versions of their programs. Within categories, for the sake of making points, evaluations are not arranged strictly by the cost of the programs (although the cost of each is specified). Many of the most expensive programs can be had for classroom use at very substantial discounts. Most companies are reluctant to reveal their "educational discount"--and none of them want to see it in print--but I can say that I have been quoted discounts as high as 80 percent.

These evaluations are cumulative--since each would be five times its present length if every major point about word processing on a microcomputer were repeated in each evaluation. Hopefully, the more common technical terms are clear from their context. Less common terms are explained in passing.

4.2 Inexpensive Programs

In one sense these are the best bargains: all of them will perform the basic functions that constitute 95 percent of the word processing work. The better ones are attractive for classroom use: even though discounts on more expensive programs can make those relatively competitive with these cheaper programs, there is still something to be said for using a simple program for the relatively simple and straightforward writing students will be doing for many courses. The very lack of features can speed the learning and use of a program. Faculty who are contemplating switching over to word processing might consider starting with one of these programs. It will not only introduce them to word processing, but will also, after some time, allow for an accurate estimation of just what additional features will be needed when buying a program for long-term use.

Personal QWERTY

Distributor: HFK Software, Inc.
Old Danbury Road
Danbury, NH 03230
617-259-0059

Cost: $99

A scaled-down version of Professional Qwerty (which sells for $325), Personal Qwerty is a new program on the market--and one that might do very well for students once a few bugs are eliminated. Certainly, its devisers are on the right track for a small program: it comes in a thin 6" X 6½" plastic case that snaps shut; its instruction booklet (it can hardly be called a manual) is a mere forty-six pages and can easily be slipped into a shirt

pocket. In short, it presents itself as a simple, budget program; its minimalist packaging makes it both very transportable and decidedly unintimidating for student users.

A number of its basic features are well-suited for students just beginning to write on the word processor: it does not use the Alt key and only uses Ctrl sparingly; the insert mode is the default, although it is simple to switch to the typeover mode; the backspace-delete key (the left-pointing arrow on the uppermost line of the IBM keyboard) is operable, making typos easy to correct as soon as they are spotted; menus appear with the most likely selection on them already designated (meaning the user only need press the return key in many cases). Underlined and boldfaced characters appear as such on the screen (albeit the boldfacing is overbright and fuzzy). And it is easy to get out of a number of difficulties by pressing Esc. Most importantly, Qwerty makes very logical use of the function keys: F1 moves the cursor one character to the left, F2 one to the right; F3 and F4 move it a word left or right; and F5 and F6 a line; F7 up a line and F8 down one; F9 searches backward for a designated word or phrase, F10 forward. Deleting is likewise logically, hierarchically organized: Shift F2 for a character, Shift F4 for a word, and Shift F6 for a line (from the cursor forward).

Qwerty has several visual peculiarities when compared with other programs--but most of them make sense and several might even be advantageous for some users. The cursor is a bright, inverted horseshoe-shaped symbol that appears under the space, rather than in it, with a rectangular box of three lines appearing in the space. When the cursor is moved backward for corrections and insertions, the rectangular box stays where the last character was typed, thereby marking the space to which the cursor must be returned to continue regular entry of text. When moved into open spaces in the text, the cursor dims and blinks. Each time the space bar is hit, Qwerty enters a dot in the middle of the "empty" space, thus making the separation of words more definitive and decreasing the likelihood of accidentally running two words together. Besides the usual items, the status line (at the top of the screen in Qwerty) reproduces whatever character the cursor is on. The screen also displays, in the upper right, a diagram of the functions its "soft keys" (the nine keys on the numeric keypad, plus CapsLock, Ins, and Del) will perform at that moment (e.g., whether striking a particular key will move the cursor left or enter a four in the text, with NumLock used to shift all the keys from one set of functions to another).

For all its laudable visual features though, Qwerty is not a screen-oriented program; not only are functions like page numbering performed via embedded commands, but so are centering and line spacing. These commands are not difficult to learn or perform, but they do result in a somewhat cluttered screen which undercuts the chief virtues of the program: its simplicity and logical organization. However, the reliance on embedded commands does mean that Qwerty can have alternate line spacings within a single document, a feature not too many of the cheaper programs offer.

Moving and copying blocks of text is very easily accomplished: the block is marked before and after using Ctrl M (or Ctrl C), the cursor moved to the spot of the insertion, and the block inserted with Ctrl I. Unlike some inexpensive programs, Qwerty does not limit the size of the block that can be moved. (It has a separate disk file to which it moves the block.) Rulers

appear on the screen--this is one case where not using a screen-oriented program can be beneficial--allowing the user to see exactly where a change in margins or tabs will take effect. Deleting a ruler is quickly done, with the text following it automatically reformatted according to the previous ruler.

Qwerty does have a few of the advanced features found in the more sophisticated programs. It will do headers and footers (only one-line ones though) and place the material flush with the left or right margin or in the center of the page. It allows for "binding spaces" (what most programs call "hard spaces"), signaling the program never to split two character units (like "A. Alvarez") between two lines. It also has a system, though not a particularly efficient one, for creating "soft hyphens," so words can be hyphenated at the end of lines and the hyphens automatically removed if a word is shifted from an end-of-line position. The instruction booklet has a very clear section explaining how to create "boiler plate" diskettes, which allow for a kind of "poor man's" mailmerging. The program can switch to an alternate font, and it permits the insertion of unprinted "comment" lines. For merging text one has to buy Qwerty Text Merge ($75). Its producers avow Qwerty will work with most spelling checkers.

The program does have a few minor weaknesses (and several major ones). For example, since typing and centering a title is usually the first thing a student writer will do, it is a bit difficult to be faced with this series of procedures right at the outset: "(1) Strike the format key, (2) Type a 'C' and a Space, (3) Type the title text, and (4) Key a Return." Justification must be turned on with an embedded command and does not take effect until printing. The Del key is used instead of the End key to finish the typing or editing of documents--a confusing redefinition, since it illogically remakes a key normally used for excision into one that performs large-scale preservations. Qwerty does not allow text to be printed via the "Print Screen" key; nor does it make use of the Home or End keys to move to the top or bottom of a file (although the booklet suggests some alternative methods for moving through a document rapidly). And the "Function Aid Card" is not a template, but only a piece of thin cardboard to slip beside and over the function keys. (If not lost, it might do for a week's use.)

One major drawback of Qwerty is that it does not allow block deletions. (They must be performed a line at a time.) For students doing short papers this limitation might not be significant, but it would be bothersome to students and faculty working on longer projects, particularly if they want to play around with alternate versions of a text by removing various large sections of it. Much more bothersome are Qwerty's difficult and very slow methods for underlining, boldfacing, and striking through. Suffice it to say that the more you perform such functions the more frustrating Qwerty can be to work with. Moreover, although it may have been a flaw on the program diskette I used for testing, underlining only seemed to "take" about 50 percent of the time and boldfacing did not always turn off on command, making these functions even slower and more distracting to perform, since the screen had to be watched very closely and some words and characters gone over three or four times to get them underlined or "unboldfaced."

The instructional booklet accompanying Qwerty is, for the most part, clear, concise, and accessible. Except for a few ruler lines, it is without diagrams, illustrations, and other visual material. Naturally, it lacks the

thoroughness of a full-sized manual, which can be an asset when trying to learn the program, but something of a deficit when trying to discover more about certain features. It does, though, devote nearly a quarter of its space to a clear, well-written glossary whose terms truly explain the functions to which they refer. The three short appendices--function keys and their arrow equivalents, format lines, and a keyboard diagram--are also good. The booklet might not tell you everything you might want to know about the program, but what it does tell you is clearly and effectively presented.

One thing the booklet will make clear is that Personal Qwerty lacks too many functions--like subscript and superscript--to be satisfactory for long-time faculty use. It might, however, be a good "trainer program," particularly for a user who planned to move on to complex programs, like PowerText and Perfect Writer, which rely heavily on embedded commands.

WORDVISION

Distributor: Bruce & James, Program Publishers Inc.
The Wharfside Building
680 Beach Street, Suite 357
San Francisco, CA 94109
415-775-8400

Cost: $49.95

For a "bargain basement" program, Wordvision has a surprising number of features, including several unique ones. While it is not the easiest program to work with--the user must first adapt to its idiosyncractic redefinition of the keyboard--its system of operation is self-consistent and logically organized. With Wordvision almost no operation is harder than it needs to be. (Moving/copying text might be an exception.)

The program is true to its word that "with minor exceptions, your printed text will look exactly like it does on the screen." For example, right margin justification and double-spacing can be set on the "Main Control Panel" before starting to type in text--and that text will appear right-justified and double-spaced on the screen. The "backspace-delete" key not only works forward as well (with Shift), but it will continue deleting in either direction if it is held down. And if the user goes too far with a deletion, text can be recalled a character at a time with the "Undo" key (the minus key on the IBM numeric keypad), a keystroke that will also undo other kinds of deletions. Formatting text is relatively easy with Wordvision, and inserting text is very easy, since the insert mode is the default. (The program does not allow overtyping, which is no great loss.)

To work comfortably with Wordvision, the user must accept absolutely the program's almost complete redefinition of the command keys on the IBM keyboard. The program renames thirty-one of these and provides a set of stick-on labels for them. (Since putting the labels on is not likely to make

you very popular with anyone else who uses the computer to operate another program, it would be a good idea to buy, for $5.00, the snap-on key caps the manufacturer sells.) The most important of the redefined keys are the four color-coded keys, which bear black symbols of the four suits in a deck of playing cards: a club on a red background (F2), a diamond on a yellow (F4), a heart on a green (F6), a spade on a blue (F8). These function as toggle switches for the most commonly employed series of commands; what each will do at any given moment is indicated at the bottom of the screen. For example, if the spade is typed when in the regular writing mode, the program will start boldfacing, and will change the designation next to the symbol at the screen bottom from "Beg[in] Boldface" to "End Boldface" as a reminder that the spade must be struck again to turn boldfacing off. (This "Beg/End" alternation is a very good way of keeping the user alert in a program where so many functions work as toggles.) Four other important keys are color-coded: Xcrpt (F1) in dark blue, Print (F9) in black, "Smart Characters" (hard space, soft hyphen, F10) in pink, with the symbol of a radiating sun, and Stop (Esc) in red in the shape of a stop sign. Other, less commonly or less importantly used, keys have light or dark gray tabs on them. These keys allow for a useful variety of cursor movements, for many kinds of deletions and erasures, for "swaps" (of inverted letters), for settings of tabs and decimal tabs, for searches and replacements, for listings of files, and for a number of other functions. In short, Wordvision may redefine the keyboard drastically, but it does so mainly to efficient ends.

Students without any word processing experience could probably adapt to the Wordvision's keyboard nearly as quickly as they could to a more "normal" one, since they would not have to "unlearn" another, very different keyboard pattern. Nor would the program's penchant for altering common word processing terms--"cursor" is "pointer", "search and replace" is "find and change"--cause any problems for the novice. And certainly the program's very low price does nothing to discourage its educational use. However, Wordvision does have the drawback of only permitting single- or double-spacing, and it is often helpful to have student writers triple- or even quadruple-space early drafts of papers, both as an encouragement to extensive revisions and as a means of allowing sufficient room to make those revisions by hand on the printout.

Wordvision has other limitations as well. Using a computer with 96K of memory limits file length to eight pages; with 256K, files can be up to fifty pages. Calculating on the basis of this difference, 128K--the basic memory capacity of the IBM PC--would allow files of only 16 or 17 pages. Another problem with Wordvision is its poorly designed "pointer" (cursor), which always appears in a "phantom space" between two real spaces, rather than on one of those real spaces. There is no difficulty with the pointer when entering text, since characters are always only to the pointer's left, but when the pointer is moved backward for insertions or deletions, it distractingly shoves characters aside to make room for itself. Moreover, as it is moved on and off a line, the characters to its right jump forward a space, then back, creating a definite visual impression that the user is unintentionally deleting characters. Besides limiting text to single or double-spacing, Wordvision makes no provision for alternating between the two in the same document. It will print out the entered text on a screen, but it will not print out the rest of the material (status line, commands, etc.), and it will not print out program screens, both of which can be had in many programs via the PrtSc key (these screens are often helpful for instructional

purposes.) Wordvision is also missing some other, not particularly important, functions, like vertical scrolling and the ability to work on one file while printing another.

On balance, though, Wordvision does very well with the number of features it does offer. One of its very best--and one that other programs would do well to emulate--is "automatic saving." At a time interval the user chooses, Wordvision will pause and save the file being worked on to disk. (It can be directed instead to emit a beep to remind users to do the saving themselves.) Anyone who has lost pages of text through forgetfulness combined with human or machine error--which is very probably anyone who has been word processing for more than a few weeks--will appreciate how useful this feature can be. With Wordvision file names can be up to forty characters--not the usual eight of most programs--and so can reflect much more completely the contents of the file. (While eight characters are sufficient for identifying files one is currently working with, they are not always enough to create a name one can identify when scanning a list of several dozen files created a year or two back.) The program allows files to be created as either "Wordvision files" or "DOS files," the latter being accessible to other programs using the disk operating system. In general, the program is orderly and efficient in matters concerning file management. Wordvision's help screens are another asset of the program: they go from a general overview of an area, to more specific explanations to more detailed explanations still, allowing users to go as far as they need to locate needed information.

Wordvision's print capacities are particularly noteworthy for a program in its class. It can print lines as narrow as 3" and as wide as 14" (and in any half-inch increments between). It will print whole files, designated pages, marked excerpts, or whatever text is on the screen. It has 255 "special characters"--foreign language characters, Greek letters, mathematical symbols, lines for drawing boxes, etc. (Of course, the printer has to support these characters if they are to be of any use.) While the procedures for invoking these characters are too slow to make working in a foreign language or with complex equations practical, procedures are efficient enough to make the occasional quotation of a foreign language sentence or the citation of a fairly simple equation a real possibility.

Wordvision has available or in the planning stage five peripheral programs ("Powerpacks...the road to growth"): a spelling checker, a thesaurus, a grammar and style check, a mailmerge program, and a program for indexing and footnoting. Presumably, these will be priced in line with the main program, so the first three, if they prove worthwhile, could be had for students at no great additional cost--and the academic writer would probably find use for the last.

Wordvision also deserves points for being perhaps the most colorful program on the market. Color-coded keys, as already mentioned, are an important part of its design strategy; on color monitors, various statuses for text are indicated by different colors, the main text being green, underlined text orange, boldfaced text blue and, if I remember correctly, superscripts purple. (This may not be a particularly utilitarian feature, but it is fun to watch.) The program manual also makes ample and generally effective use of color--with many color illustrations of the color-coded keys, a goodly number of sample screens (in green or in gray with blue highlighting), yellow boxes

containing important instructions, several four-color diagrams, arrows and error messages in red, and a color code for each of the twelve chapters and the appendices. The manual's graphics are, in the main, also good. The writing, on the other hand, is hardly scintillating, and there is not too much continuity from point to point--but the tone is rarely condescending, smartalecky or even, despite the phrase on the "Powerpack," quoted above, unduly enthusiastic. The manual's "Whirlwind Tour" probably covers too much material to be an effective "quick start course," but it does allow the user who goes through it carefully to take the "Grand Tour" which follows fairly rapidly. The error messages also include suggested corrective actions. The box that the program comes in folds into a triangular stand into which the manual can be slipped so it can rest upright next to the terminal and be flipped through with one hand. Finding what one is looking for in it is made easy by a thorough index and difficult by the lack of chapter tabs for quick location of subjects.

Wordvision is one of the programs not likely to stand out in a "checklist review," since it lacks a number of features which are usually surveyed and has a number of good ones that are not. Besides, it is not quite fair to compare Wordvision with programs costing ten or twelve times as much. But it merits consideration for use in the classroom, for use by academics who do not have complex word processing needs, or even by academic writers who are looking for an inexpensive second program, one that will easily allow them to do that share of their writing that is simple.

MEGAWRITER

Distributor: Megahaus Corporation
5703 Oberlin Drive
San Diego, CA 92121
619-450-1230

Cost: $99.95

MegaWriter touts itself as a full-featured alternative to much more expensive programs: "We give you all the features you'd expect to find in a quality word processor [and]....We do it all for $99.95!" Unfortunately, as this phrasing unwittingly suggests, MegaWriter is not a "quality word processor."

The main (and most pervasive) problem with MegaWriter is its clumsy and confusing overall design, which makes it hard to learn and uncomfortable to use. It tries to be mnemonic by using letter-identical keystrokes for commands, but undercuts the value of this strategy by representing the commands, in the manual and on screen, with a distracting open parenthesis in them--e.g., "S(ave" and "Q(uit." The user instinctively waits for the closing parenthesis, which never comes, and ends up having to make sense out of lines that look like "O(n-line - list disks on-line." Although the open parenthesis is supposed to segregate keys to be struck from the rest of the characters on

a line, this distinction is sometimes blurred: e.g., the manual represents the superscript keystroke s as "X(+" and "X(-" whereas they should be "X+(" and "X-(" using the (formula. Perhaps just closing the parenthesis would alleviate most of the confusion created by this device.

More difficult to correct will be the poor design of the "Write Mode." When you enter it from the main menu, you are still not ready to "write" (i.e., to start entering text), but must first choose one of the functions from this bewildering command line: "Write: A(djust C(opy D(elete F(ind I(nsert J(ump R(eplace Q(uit X(change ?" By MegaWriter's odd definition, to write is to insert, so "I" must be pressed. Furthermore, the status of normal text entry is not the default, but just one of nine equal choices. Thus, if you stop inserting text to perform any other function on the command line, you are not automatically returned, as you would expect, to "I(nsert" so you can get on with typing in material, but instead are left "nowhere" and must press "I" again. If you forget to do this and start typing text, nothing will happen until you come to an a,c,d,f,i,j,r,q, or x, which will be taken as a command--leaving you, at best, puzzled and frustrated about what then happens on the screen and, at worst, faced with a mangled text.

Like the series on the command line, many functions in MegaWriter seem much more complicated than they need to be. For example, instead of being able to shift to an "overstrike" status, you must employ the "X(change" command: to replace "Fred" with "Brad" (this is the manual's example), you cannot simply place the cursor over the "F" and type the new name, but must instead invoke the "X(change" command, then type "B-r-a-d" and then hit the "End" key. Working with (and remembering) commands on the command line is even more difficult after it disappears, which it will do once a screen of text has been generated. Similarly, the absence of end-of-paragraph symbols makes it difficult, at first, to figure out why and how the program keeps readjusting your text. And there are another two dozen of MegaWriter's operations which the user--if my experience is at all typical--will find equally confusing, illogical, and difficult to recall or deduce.

Nor is the manual likely to be a great help. A spiralbound book, it has the look of a "cheap" publication, with typed rather than set text spaced a line and a half apart. The manual lacks visual accentuation, proportional spacing, a variety of typefaces, and any number of other features readers have come to expect in printed books--and whose absence they very much notice. Considering how much material must be explained, the manual needs longer and clearer descriptions, more illustrations (and not just of sample screens), more readable (and better highlighted) charts of commands, and a more thorough index.

The manual has a number of typos in it, a few of them rather distracting--is "D(lete" (p. 12) a truncated word like "X(change" or simply a misspelling? (It turns out to be the latter.) The manual and other accompanying documentation are full of, for me, infuriating parenthetical comments: "(oh yeah!)," "or (gulp!) a bad printer," "(lucky you!)." The overall tone is relentlessly encouraging and upbeat: "Ready, Set, Print!" and "Now for the real magic." Naturally, the manual has some positive features, like the tree diagrams of the programs' three modes (Write, Print, and Organize), and despite its poor writing, the manual does manage to explain the program's functions in fairly logical order.

To be fair, it should be said that MegaWriter does live up to its claim of providing many "features you'd expect to find in a quality word processor." It has a delete buffer, a separate Organize mode for convenient file management (backing up, deleting, renaming files, even formatting disks), a "Zap" command for large-scale deletions (with a built-in warning on deletions over eighty lines), an embedded command that will create alterations in line spacing, and a series of commands that will move the cursor around and about a document quickly. Its paired use of "End" and "Esc"--to enact or cancel a command--is particularly sensible and easy to remember. MegaWriter also has the capacity to copy material from one file to another if both sites are on the same disk; it will create headers (but not footers); and it can be customized for a variety of printers and will do microjustification if the printer supports it. MegaWriter avoids, at some cost, the Alt and Ctrl keys in its command sequence (a benefit for users who find these "extra" keys disconcerting), and it reserves the function keys for "Macros." These keys can be programmed to perform functions and tailored to and stored with a particular document, and Macros can be created afresh for each new document. A good number of even the most powerful programs do not have this capability, and the ones that do advertise it prominently.

MegaWriter would probably be a terror to work with in the classroom, but it might be worth buying for the academic writer with complex word processing requirements and more time and patience to expend than money to spend.

TEXTRA

Distributor: Ann Arbor Software
PO Box 7345
407 N. Main
Ann Arbor, MI 48107
313-769-9088

Cost: $95

In a number of ways, Textra is what MegaWriter tries to be; namely, a competent, fairly easy-to-use program, which keeps most of its commands away from the Ctrl and Alt keys, offers a few of the specialized features usually found only in more powerful (and much more expensive) programs, and, in sum, can be said to give more value per dollar than programs in higher price ranges. With Textra you get your $95 worth.

Perhaps Textra's strongest selling point is its excellent on-line tutorial, which helps turn the program's method of operation--in itself neither particularly splendid nor awful--into a comprehensible system. Although not interactive in a literal sense (it gives no tasks to the learners), Textra's tutorial very sensibly allows users almost complete control over the speed, content, and direction of its presentation. Normally, instructional text appears in a box on the upper right of the screen, one line at a time, at a pace geared for slow reading, and remains on screen for a few

moments after it fills in. Always on the left side of the screen is a list of the seven options users have to alter this pattern: "Go to Table of Cont., Exit Tutorial, Restart section, Pause, Fast Forward, Slow down, Speed up." (Only moving backward within a section is not permitted.) Instructional material is in short, declarative sentences that link to what has gone before and what is to come; it is well-spaced, with points always separated by at least one empty line. Many screens also have illustrations (of sample screens, of various sections of the keyboard, of lists, etc.), which appear, clearly set off, on the bottom of the screen. The tutorial has, to good effect, many more screens that most other tutorials. The screen that introduces the important concept of Textra's tripartite insertion strategy is a good example of the tutorial's clear, uncluttered presentation:

> As was briefly mentioned before, there are three ways to insert text.
>
> The first method is very simple - move the cursor to where you want to insert text, and type away.
>
> The second method is called "compose mode." It is used when you want to insert one or more lines of text.
>
> The last method is called "overstrike mode." Characters typed using this method will replace the character at the cursor.
>
> Let's take a look at each method.

Textra is evidently based on two organizational notions: 1) that most commands should be handled via a series of menus and sub-menus which employ the ten function keys and 2) that many decisions about formatting need not be made until the document is readied for printing.

The main menu appears, at the bottom of the screen display (separated from the entered text by only one line.) While not especially clear visually, conceptually it is not difficult to learn. The main menu looks like this:

F1 menu F2 comp F3 reform F4 label Del-- F5 F6 F7 line F8 eol CF9 top CF10 bot

Pressing F1 will call up the command menu:

Main 2 Shift 3 Search 4 Replace 5 Copy 6 Move 7 Print 8 Switch 9 Merge 10 Exit

Hitting F3 to F9 will give you a sub-menu from which to choose the exact function you want. This pattern is made even more sensible by consistent formats. For example, "exit" is always the first choice in a sub-menu, so it is easy to "retreat" if you find yourself in the wrong area. Moreover, there are shortcuts for some functions; like the more expensive programs, Textra is

good at giving more than one way to do a number of things, so experienced users need not go to the sub-menu to complete a function. This menu/sub-menu schema permits Textra to avoid using the Alt key for commands altogether and to reserve the Ctrl key for a limited number of large-scale functions, such as deleting from the cursor to the bottom of the screen and moving the cursor to the beginning of the file.

Textra's tendency to "postpone" operations until getting ready to print is exemplified by the fact that it contains a command (and wisely includes a Y/N? double-check to confirm it) that allows the user to reformat an entire file at once after an editing session. (Reformatting can also be done paragraph by paragraph.) It is as part of the preparation to print, that line spacing (single, double, triple) is chosen for the document. Textra has a "page preview" command also as part of print preparations: this command puts the document on the screen as it will appear in print and allows the user to spot places where a hard page break (one of the program's few dot commands) or some other adjustment is needed to improve the look of the piece. Clearly, Textra sees the entered text as "raw material" which must be "refined" before it is ready for printing.

Textra's peculiar liabilities and capabilities are about evenly matched, except that several of its limitations might immediately preclude its use by those with certain requirements or by those who cannot adjust to its design idiosyncracies. Most importantly, it cannot create very long files, a limitation the manual tries to disguise by invoking large numbers: "Systems with 128K of memory will have a buffer ranging from 40,000 characters (with DOS 2.0) to 60,000 characters (with DOS 1.0 and 1.1). Systems with more than 128K of memory will have buffer sizes of 60,000 characters regardless of the DOS you are using." What this means in technical terms is that the program cannot do "swapping," the function whereby files longer than the computer's active memory capacity can be created, because sections of the file can be "swapped" to a disk, then recalled if needed (which is why, for instance, it takes a "swapping" program a few moments to enact a "go to beginning of file" command when you are, say, on p. 51 of a document). What this means in practical terms is that with Textra your document can never be more than about eighteen pages, and may have to be as short as twelve.

Textra has several design weaknesses. The most glaring is that when largish amounts of text are inserted in a line, the words at the end of the line are not automatically moved down to the next, but are pushed beyond the right edge of the screen (with a high-intensity "&" inserted to note this). These lines can be scrolled to horizontally, but since most users will find it hard (if not impossible) to work with part of their text invisible, insertions of any length often mean stopping to reshape and reformat the text, a bothersome task. Another noticeable weakness is that Textra, for some reason, has no command for deleting single words--they must be gotten out a character at a time. It cannot do alternate line spacing either.

Some users will find the program's screen too skimpy for their taste: it lacks a status line, has no box, lines, or other indications to mark the "page," and, as already noted, separates the main menu line from the text only by a blank line. Some users may also find the use of the Ins key to switch to the overstrike mode a distraction, especially if they have used programs when "insertion" and "overstriking" are opposed modes. The program can only do

headers and footers in a limited way: i.e., one of each, of no more than thirty-nine characters, can be specified on the print menu, with header/footer placement right, left, centered, or alternating. (Note, however, that page numbering is a separate operation on the print menu.)

For every clumsy (or worse) design feature in Textra there seems to be a clever and useful feature to balance or compensate for it. For example, the ways of saving (or not saving) are varied and all useful:

F1 - Exit without saving changes

F2 - Save with control characters
F3 - Save value of switches

F4 - Save current work, then edit new file
F7 - Save with same name
F8 - Save with new name
F9 - Save current work to disk, then return

F10 - Return to the editor

This menu allows for the easy saving of all drafts (just use F8 each time), and makes periodic saving of the document being worked on a matter of only two keystrokes. Files can be saved with all their format specifications. Using F7 automatically turns the prior version of a file into a backup file. (While some reviewers of word processing programs think backup files an "extravagant waste of disk space," I will attest to their usefulness: more than once I have gotten a "bad sector" on my primary file and would have lost the whole document without a backup version.)

Another notable feature of the program is that it offers a separate program disk file (TXCUSTOM) that allows the user to customize Textra to his or her liking and regular uses--e.g., making 5 and 75 the default margins, having boldfaced characters struck five times, adapting the program to the printer. This customized version of Textra is then saved as "your" permanent one.

A helpful design feature for editing is the program's "compose mode": for insertions of a line or longer, F2 on the main menu can be pressed, which moves the following text down two lines and makes the insertion easy to see, since it is not shoving text to the right as each character is entered. The program comes with a number of auditory signals built in, like the two beeps emitted when a new file is started; users can turn them off or intensify them as they wish. (In fact, Textra allows a significant number of features to be turned on and off.) For those who find them more comfortable, there is an alternate set of cursor control movements to the arrows: Ctrl + H,J,U,N. The PrtSc key works with all of Textra's screens, including those in the tutorial.

The most complicated thing about Textra might be its user's manual, which is oddly opaque in its explanations and rather choppy in its writing (especially in light of the simple, effective writing in the tutorial). It cannot seem to make up its mind whether it wants to be a step-by-step guide or a reference guide, and succeeds at being neither. Its index is radically incomplete, although it is followed, somewhat helpfully, by a page reference

listing of (some) commands. At a few points the manual even seems in despair over how it has been going about things ("enough lecturing!"). Its three pages of error messages deal exclusively with input/output errors (i.e., things like "file not found" and "error renaming file"). However, the manual is complete--and the persistent will sooner or, more probably, later find what they are looking for. Unlike the manuals accompanying the other inexpensive programs reviewed here, Textra's is a looseleaf binder, which allows for the insertion of revised pages. (Like the producers of more expensive programs, Textra's producers regularly issue new pages that correct errors, refine explanations, and add details.) The quality of the print is high, with reproductions of "screen text" well highlighted by blue print; the glossy paper looks like it will withstand heavy use.

Textra is a program that will appeal to some users more than others, both in terms of what it can do and how it goes about doing it. In its price range, it is a "safe bet," although probably not the best one for most student users. Because of its odd mix of limitations and sophisticated features, it might serve well for some kinds of faculty and (particularly) administrative writings, and not at all for others.

PFS: WRITE

Distributors: Software Publishing Corporation
1901 Landings Drive
Mountain View, CA 94043
415-962-8910

Cost: $125

PFS software is designed along "divide and conquer" lines. PFS: Write is the word processing component in a software network that includes PFS: File, PFS: Report, PFS: Graph, PFS: Solutions (twelve sub-programs) and PFS: School Recordkeeper (four sub-programs). The first three of these can be used with PFS: Write for, respectively, creating form letters, incorporating reports in other documents, and adding graphs to documents. "What could be faster and easier than PFS software?" asks the company's sales brochure. "'Faster' or 'easier' than what?" we are tempted to reply, archly.

PFS: Write is one of the slicker low-priced programs, well-packaged and well-documented. Like all the PFS software, it seems designed primarily for business and administrative functions. For example, it has a special line-spacing designation for printing addresses on envelopes.

The program's screen space is well-defined, with the text boxed and an easy-to-read status line beneath it. Formatting commands are reflected, in most cases, by changes in the text on screen (although text always appears single-spaced there). Error messages appear on the status line, where they are effectively segregated from the body of the text and easy to read. Because the program activates the "Print Screen" key, the help menu, which is

not one of the best, can be printed out after it is called up, so inexperienced users have a ready reference if they find alternating between text and help screens distracting. Being able to print out copies of the help menu is also useful for introducing students to the program before they start to work with it on the computer. While the insert mode is not the default--one must press Ins, which turns the cursor into a blinking underscore--PFS: Write does not, like some programs, automatically shift out of the insert mode when the cursor moves to a new line or new paragraph. Nor do insertions necessitate reformatting, since text is pushed forward a character at a time and lines are readjusted accordingly.

The PFS: Write manual is one of the easiest to use, most particularly because it relies heavily on illustrations: of screens, of sample documents, and even of short sections of text as they would appear on the screen (illustrating, for instance, that typing over "condominium" with "hotel" would result in "hotelminium," meaning six characters would still have to be deleted). Very handy are the summaries of major points at a chapter's end, especially those summaries in which commands are accompanied by pictorial representations of the key(s) struck to effect them. Appendix B is a useful (and not at all redundant) compilation of these chapter summaries, as Appendix A is of various error messages, their meaning in plain terms, and the "corrective action" each requires noted. The other three appendices deal clearly with "Useful DOS Commands," adapting the program to a variety of printers and video monitors, and installing it on a hard disk. The glossary is clearly written but too skimpy (2 pp.). In general, the material that is summarized and repeated is done so in effectively brief fashion and allows the user, once basically familiar with the program, to find information quickly and to avoid having to reread longish sections of text. The program comes with a "sampler disk," which contains fifteen documents for use with demonstrations and exercises in the manual. While nothing special, they do aid the user in learning the program a bit more quickly than creating his or her own practice documents would.

The clarity and reasonableness with which PFS: Write is presented can obscure some of the program's limitations. For example, one must read very carefully between the lines of the program manual to discover that it is not possible to create files longer than about eighteen pages. (Additional material will have to go on a new file.) You will search the manual in vain for subscript and superscript commands--the program cannot perform these functions. Sometimes, the program operates too literally, e.g., putting on the screen as blank space the twelve lines it skips between the last line of one page and the first line of the next, thus leaving distractingly large "holes" in the text as it appears on the screen.

A distinct disadvantage of PFS: Write is that it requires consistent line spacing throughout a document it is printing--and that spacing can only be single or double (or the aforementioned "envelope spacing"). Since, as has already been noted, it is often very important to triple- or even quadruple-space documents which will be heavily edited and reworked by hand when printed--especially early drafts of student papers, this line spacing limitation is a significant drawback.

Several of PFS: Write's design features can be somewhat difficult to adjust to. For example, F10 and not Return is the keystroke to "continue" in

most situations. And since end-of-paragraph symbols are not recorded on the screen, shifting them around and moving material between paragraphs is made more difficult thereby. Nor is the program's main menu particularly well-designed. For instance, Selection 1: "Type/Edit" is not what you would choose to call up a previously created document for editing. The problem with the six selections on the main menu is that they are not discrete and must often be employed in tandem: to print a document (3), one must, unless it is already in memory, first type 4 ("Get/Save/Remove"), then 1 ("Get") under the subchoices, then the document name, then F10, and only then 3 to start the print process.

Some of the program's advantages and disadvantages are closely allied. Headers and footers are easy to create and appear on the screen just where they will be on the printed page, but for some reason they are always centered and cannot be made flush with the left or right margin. The fact that many commands, like underlining and boldfacing, are enacted on the screen makes the fact that double-spacing is not all the more anomalous. Block moves are made more efficient because the text marked to be moved appears highlighted in a very visible solid color block, but they are made less efficient because only thirty-five lines can be moved at once due to the small size of the "block buffer."

Like many programs, PFS: Write seems designed with essentially one kind of document in mind. The prime model for this program is apparently a simple to moderately complex business document, of no sustained length, but often likely to have included in it material created by allied programs. Also, the program would seem to assume as its typical user a fairly inexperienced (and somewhat computer shy) secretary/typist. Consequently, PFS: Write is a program with many safeguards. It warns the user about trying to retrieve another file without first saving the one in memory; it always questions an "overwrite file" command (Y/N?) before enacting it; and it offers an easy way of preserving earlier versions of documents by allowing an alternate name to be given the reworked document as part of the procedures followed to save it. Likewise, the program uses Shift F5 and Shift F6, rather than the simpler F5 and F6, as the commands for deleting a word and a line, thereby making accidental deletions less likely to occur through an errant keystroke. (Logically, Del is used for character deletions.) There is a simple command for checking the available disk space to see if there is room for the file about to be created, and the program has the ultimate (or is it penultimate?) safeguard of permitting the entire contents of one disk to be copied quickly to another disk. The user who forgets a file's exact name can call up a file directory of the disk in use as part of the "Get/Remove/Save" procedures.

Cursor movement in PFS: Write is as fast and flexible as with almost any program: arrow keys for one space left or right, and one line up or down; F3 to move to the previous word, Shift F3 to go to the beginning of the line--with F4 and Shift F4 for going to the next word and the end of the line; the PgUp key will move the cursor up a screen of text, PgDn down a screen; the Home key moves the cursor to the beginning of a document and the End key to its end.

For an inexpensive program, PFS: Write is noteworthy for the number of print options it offers: the ability to print any designated set of pages from a document, or just one page; page pausing for single sheet printing;

multiple copies; automatic indenting (to adjust for printers which place text too close to the left side of the page); and the chance to designate other documents to be included in the one being printed out. In addition, the program is designed to take advantage of a number of special features--like double-wide mode--on a variety of printers.

Because it is part of a network of programs, PFS: Write is particularly adept at incorporating material created by its allied programs. (Obviously, you have to buy these other programs to take advantage of this attribute.) With proper preparation, PFS: Write can even incorporate material from non-PFS programs. Another very sophisticated aspect of the program is its search and replace capacity. Besides the normal searching out/replacement of words and phrases, it can also, like many of the more powerful programs, do "wild card searches"--i.e., it can search out partial words and phrases. Thus, it could find every word starting with "abs..." or ending with "...ies" or even every word with "...iti..." anywhere in it.

All in all, PFS: Write is an easy-to-use, screen-oriented program with some very basic limitations. To the extent one's regular work, in the classroom or the home office, produces documents that resemble those regularly produced in a general office situation, PFS: Write is a program worth investigating.

SELECT WRITE

Distributor: Select Information Systems
919 Sir Francis Drake Blvd.
Kentfield, CA 94904
415-459-4003

Cost: $99

Select Write is the diminutive version of Select: The Word Processor (which sells for $295), but it contains very many of the features of its progenitor, which for several years has been one of the more popular word processors on the market. Although it lacks some of the advanced editing features that distinguish its competitors, Select Write probably offers the best balance of basic and advanced features of any program in its price range.

What is often said, in so many words, about Select: The Word Processor holds true for Select Write as well, namely, that it is an easy program to learn but a difficult one to use. What makes it easy to learn is that it is an English-language program which virtually dispenses with symbols and which entirely avoids the Alt, Ctrl, and (except in a very minor way) function keys in its command sequences. No other program is so immediately understandable on the screen--and few other programs are supported by such clear, helpful documentation (including a very accessible, if annoyingly friendly and encouraging, tutorial). Select Write's main menu is a good illustration of how much complex material can be presented with perfect clarity through its English-language command structure. Even a first-time user would be able to

understand, and so exercise, these options. (Notice, for instance, in the following illustration that there is a clearcut distinction between "creating a new document" and "editing or changing" an old one, a distinction many programs do not observe.)

>SELECT WRITE: Create Edit Print Help Teach Quit List View Delete Name Alter

CREATE	- To Create a new document
EDIT	- To Edit or change a document
PRINT	- To Print a document
HELP	- To get an explanation of a command
TEACH	- To learn how to use Select Write
QUIT	- To Quit your Select Write session
LIST	- To List your documents
VIEW	- To View a document without changing it
DELETE	- To Delete unwanted documents
NAME	- To change the Name of a document
ALTER	- To use Select Write in special editing mode

Subsequent menus are organized along the same clear pattern of single letter keystroke commands.

What makes Select Write hard to use is that it is not screen-oriented and, more significantly, that editing remains a clumsy, slow procedure no matter how proficient the user becomes with the program. Select Write is based on the premise that writing and editing/revising are essentially discrete acts--a notion much writing research does not support--and so the program does not allow for the easy switching between the two functions. Further, it sees editing changes themselves as discrete acts, each of which must be ended (by pressing Esc) before the cursor can be moved to the location of the next editing change. Those with very orderly revision habits or those who make all their changes on a printout (and only need enter them) will adjust well to Select Write's editing pattern, but the majority of writers, at all levels, who engage in trial-and-error revising and rewording as they write will find that this program inhibits them. For example, if you notice a sentence three lines up that needs a qualifying adverb in it, this is what you must do to put it there: exit the insert mode in which you are entering text and go to the editing menu (via Esc), set the cursor insertion to backwards (<), move the cursor using the arrow and other keys (the cursor can only be moved, except by backspace-delete, by going to the editing menu) to the spot of the insertion, re-enter the insert mode (which then clears a space of two lines), type the insertion, press Esc, set the cursor direction to forward (>), move the cursor back to where you were, and re-enter the insert mode yet again. (Designating "insert" as the mode wherein text is both created originally and added to existing text is one place where there is a somewhat confusing overlap in Select Write's normally clear system of terms.)

Needless to say, such a regimented, complicated series of maneuvers discourages implementing those changes that pop into one's mind for a moment

and are likely to be lost if not quickly recorded. Since such changes can, on a word processor, be eliminated or further refined so easily, it is usually a definite loss not to get them down while they are in mind. Programs that have (or can set) the insert mode as the default provide a great advantage in this regard. Even ones that employ the Ins key as a toggle much facilitate the process of recording one's momentary ideas for later rephrasings and improvements. Select Write definitely does not facilitate this important writing activity.

With Select Write, erasing and performing a number of other functions are as complicated and potentially offputting as making insertions. Users can, because of the program's clear writing and excellent documentation, learn such functions quickly--but they will never be able to perform them quickly.

A special category of Select Write commands are its embedded "back-slash commands," which give the program many of its noteworthy capabilities, but which also account for much of its lack of screen orientation. (The facing versions of a simple letter, as printed on the left and as it appears on the screen on the right, on pp. 100-101 of the manual, demonstrate just how non-screen-oriented these commands make Select Write.) Backslash commands, like "\E" for a hard page break, are inserted in the first two (or three) spaces on a line--after which no text can be typed, except headers and footers, since the program reads everything on this line as a command and does not print the line. To make backslash commands possible on every line, the cursor always drops down to the first space on a line when you are typing in text, then jumps ahead to the normal left margin if any key but the backslash is struck. This whole method takes some getting used to--with the cursor jumping forward ten spaces each time you start a new line of text, with words lost because a carriage return was not inserted after a backslash command. But it is not so much the method as the look of the screen, both cluttered with command symbols and full of open spaces after them, that is a continuing annoyance for most users. However, the backslash command system is very useful for creating documents with complex and variable formats--including, of course, as many variations in line spacing as needed. Select Write also allows for the printing of a draft version of a document, which prints the document exactly as it appears on screen, backslash commands and all (a very useful way of discovering where commands were missed--like the direction to turn off centering--when the regularly printed version of a document does not come out as expected).

Besides a reliance on embedded commands, Select Write resembles some of the more expensive programs in other sophisticated ways. Not only does it offer five "text enhancements"--boldfacing, underlining, overstriking, subscripting, and superscripting, but it also allows them to be used in conjunction with one another, a capability you will not find in other $99 programs. Select Write has a good variety of cursor movements, some of them unusual (e.g., in the editing mode, pressing a comma will move the cursor backward or forward by comma). Erasures can be done by character, word, sentence, screen, or block (although not by line, unless it is marked as a block first). Select Write's "Append" command will insert one whole file into another; its "View" function allows one file to be looked at, although not in a "window" but on the whole screen, without interrupting the editing of another; its "Write" command creates a new file and moves to it a block of text you have marked. And cumbersome as it is, editing can be made a bit

easier by turning off the on-screen justification while you are doing it, to avoid the distraction of the screen constantly reshaping itself.

As important as all these assets are, together they might not be as important as the fact that Select Write does not limit the size of a file, except by the available disk space (about 100 pages on an otherwise empty 5 1/4" disk). Moreover, it warns the user of any impending space problems, like the fact there will probably not be room to edit a file on a disk that is almost full.

Another benefit of Select Write is its on-screen "TEACH" tutorial. Not, to my mind, quite as good as Textra's, it still has several advantages over that one: it is on the program disk, and so can be accessed directly from the main menu. Divided into twenty-seven sections, each a "How to" ("insert," "read the command line," "format," "move," "zap," etc.), it serves both as a two-hour introduction to the program's more basic functions (most followed by a short exercise) and as a continuing reference, any section of which can be called up to the screen with a few keystrokes. And, the tutorial is neatly complemented by the program's "Help" command, which will supply an explanation of any command that is typed in after the help key is hit. Where it falls short of Textra's tutorial is in providing an overview of the program's logic and organization.

That lack, however, is largely compensated for by Select Write's manual, a hardcover book with a glossy slipcover. (It has the look of a text you might have used for "Second Year French.") The manual succeeds, as Textra's does not, in being both a step-by-step "walk through" of the program and a reference guide to which users can return to review functions and learn more about them. Its index is excellent, not only full, but often containing more than one listing for an item. For example, locating material on "double spacing" can be done by looking under either "double spacing" or "line spacing." It may take a little time to get used to the manual's organization. For example, the chapter on creating and editing a document discusses commands, save for the all-important "Insert," in alphabetical order rather than in the order of their likely use or of their importance. However, the manual will clarify any of few points the program's generally clear editing screens, help screens, and tutorial screens do not.

Although Select Write's manual will appeal to students--not in the least because it resembles in shape and size some of the texts they are using for their courses--overall the program is not the best one for most student uses, particularly for students beginning word processing. What Select Write teaches so easily is its whole operation--and students in most courses do not need to learn half this much. What they do need to learn, most particularly the program's cumbersome ways with insertions and deletions, will not encourage them to make the best use of the special capacities of editing/rewording/revising that are the greatest benefit to students using word processing. For many faculty with complex word processing requirements and who work almost exclusively in English, however, Select Write will be the least expensive program that can meet all their needs. If academic writers can eventually adjust to its awkward editing procedures and screen format--and it probably takes about fifteen hours' work to see if they can--Select Write will probably do very well and very completely and very cheaply for all their tasks.

4.3 Moderately Priced Program

Although it is difficult to generalize, moderately priced programs can be expected to provide several kinds of advantages over their lower-priced competitors. First, they will have more functions and features, and often will have alternate ways of performing a function. Second, they will not have most of the limitations of the cheaper programs, most particularly limitations of the size of files (except, of course, as dictated by available disk space). Third, many will be able to do combined functions, like underlined and boldfaced type. Fourth, they may have "help screens" which explain functions in some detail, and not just "help menus" which outline them. Fifth, a number come with on-screen tutorials or have extensive tutorials (including exercises for learners to do on the computer) as part of the manual. Sixth, those manuals tend to be fuller and better produced, often as looseleaf binders, so updated pages can easily be inserted. Those manuals often also include a full reference guide, usually alphabetically arranged, following the chapters outlining program operations. These programs are probably adequate for the kinds of writing most English faculty do--and the writing students do in English classes. On average, they will not have as many features as expensive programs--and may be lacking just the features a user needs for his or her particular kind of work--but at least some of them can stand as relatively "full-featured" competitors to more expensive programs.

VOLKSWRITER DELUXE

Distributor: Lifetree Software, Inc.
411 Pacific Street, Suite 315
Monterey, CA 93940
408-373-4718

Cost: $295

Volkswriter is a very reasonable and sensible program--and apparently a popular one with owners of IBMs who want an "intermediate" word processing program. It is also a program which demonstrates how using Alt, Shift, Ctrl, and the ten functions keys need not necessarily make the command structure confusing or commands difficult to find or recall. (Some programs go to extreme, unproductive lengths to avoid having to use some, or even all, of these keys in command sequences.)

One of Volkswriter's great strengths is the overall clarity with which the program is presented. Unlike many programs, it begins at the beginning. The manual, if somewhat self-congratulatory, is very easy to follow step by step (and rarely necessitates jumping ahead for more information). The "Function Key Reference Guide," which F1 (the "Help" key) puts on the screen, is the most complete and concise thing of its kind in any program: in a boxed chart of only six lines, it displays the forty different operations the function keys perform (by themselves, with Alt, with Ctrl and with Shift), and it even reminds the user of the existence of the other command keys (the four

arrows, Ins, Del, Home, End, PgUp, and PgDn). As this on-screen guide illustrates, Volkswriter does not assume the reader knows where to start looking for a desired function, but instead lays many out at a single keystroke--an attitude any inexperienced (or forgetful) user will much appreciate.

Volkswriter is obviously a second-generation microcomputer word processing program, one which draws on the good features of a variety of earlier programs, while trying to avoid their weaknesses and to add a few of its own unique features. Certain of the features it "borrows" will affect most users neutrally; e.g., the text is moved up half-a-screen when you come to the bottom of the screen as you enter text. Most users will approve of the straightforward status line, which indicates the line, column, and page where the cursor presently is. A few of its "wrinkles," like the fact that multiple copies are printed via an embedded command, may seem odd (although not necessarily inefficient). Any of its design features (e.g., the F8 which reformats text from the cursor to the end of the paragraph) will seem sensible and utilitarian. A few semi-standard features, though, will be found wanting; e.g., Volkswriter's designers did not think the "soft hyphen" feature worth including (and they may be right).

The manual stresses the program's "forgiving nature"--clearly its makers wanted to avoid the problems with lost text, lost files, even erased disks that one could stumble into in some older programs. F1 is, at all junctures, the "Help" key, as F10 (or Esc, a useful doubling) is the "reset" key (what the manual likes to call the "panic button"). Many of the program's larger functions carry strong built-in warnings. For example, a simple command whereby you can store a file on a different disk than the one from which you retrieved it (or even put it on one disk after another) very demonstrably warns you not to remove the original disk if you are trying to perform this function with "a very large document," one too long for the computer's active memory. (See comments on "swapping" in the section on Textra.) In this and virtually all other situations, the user must be heedless to fall into large errors concerning file management.

Although it lacks a fair number of (not usually important) features--like spooling, double-column print capability, "wild card" searches, and a delete buffer--Volkswriter has few design weaknesses per se. Nonetheless, two aspects of its design might strike some users negatively. First, it is what might be termed a "middle state" program. It is neither wholly screen-oriented nor wholly print-oriented. Certain features, like boldfacing and underlining, will be enacted on the screen display, while others, like double-spacing and right justification, will not. (Volkswriter does offer Ctrl F3, which lets you see your text with all directional signals removed, as a partial compensation for this incongruity.) The program does not require going to an edit mode or an edit menu to make insertions, but neither is "insert" the default. The Ins key, which turns the cursor from a blinking underline to a blinking rectangle, must be turned on and off--and it cannot be left on, since it shuts off automatically when the cursor is moved to a new line, returning the user to the less useful overstrike mode that is the default. (Alt F3 complements Ins by allowing for the insertion of a whole line.)

Second, Volkswriter is a rather eclectic program, getting its jobs done in any way it can, rather than in a schematized order. For example, shifting from double- to single-spacing is not done directly; rather one must embed a "..Format" command which directs the program to start printing according to a different format, one that designates single-spacing. To go back to double-spacing, the original format command would have to be reinserted. (While a rather oblique procedure, and one that calls for the premaking of formats, it does allow other changes, like the shifting of margins, to be allied with changes in line spacing.) In fact, Volkswriter very much expects users to create any variety of regularly employed formats--and not only permits formats to be stored on document disks, but also suggests the most commonly invoked ones be stored on the program diskette as well. In this respect, the program resembles, chiefly to good ends, one variety of word processing programs. In its general avoidance of menus in favor of direct, single or double keystroke commands--to be sure, it has a "Command Menu" and a "Format Selection Menu" and a few others--Volkswriter resembles another, different variety of programs. Thus, experienced word processors may experience deja vu in working with Volkswriter, and may experience it in several conflicting, perhaps disturbing ways.

Besides having the usual features one would expect from a program in its class--e.g., printing any designated part of a document, creating variable headers and footers which can be placed right, left, center or alternating, and numbering pages in a variety of ways--Volkswriter has several advanced features worthy of note, and one that is exceptional. Its "boilerplate" capacity is subtle and useful--and very carefully explained and illustrated in the manual. For more extensive mass mailings, there is a built-in "textmerge" program. The automatic creation of back-up files is a function that can be turned off and on, a feature which should satisfy both sides on the question of whether backup files are a waste of valuable disk space or an important "safety" feature. Volkswriter's "notepad" function allows for single-line notations to be written while working on a document and immediately removed and collected together for later retrieval. Underlining, boldface, shadow printing (darker than boldface), and strikethrough shut off automatically at the end of paragraphs, subscripts and superscripts at the end of lines--which means the absent-minded (like myself) cannot end up with strikethrough pages of underlined or boldfacing or text if they forget to turn off a function.

Most exceptional is Volkswriter's foreign language capability, allowing the user to write easily in French, Spanish, Italian, and German (or any combination of these and English). The Alt key is used in conjunction with the numbers and letters to produce all the needed non-English characters. For example, Alt with 1,2,3,4,5 produces â,ê,î,ô,û. Alt 6 produces the British pound symbol and Alt m the inverted question mark, and so forth. The manual's appendices also contain explicit directions for using the program with IBM machines designed for the French, Spanish, Italian, and German markets. Academics who work in these four languages will most certainly want Volkswriter as their word processing program.

The Volkswriter user's manual is surprisingly good, especially in light of its unpromising look: a looseleaf binder, with readable but not high quality type (or paper), it lacks illustrations except for some unhighlighted lines of screen text, and it covers program operations in what appear to be four skimpy chapters. But better than any other, this manual presents just

the right amount of information, including the most basic, in almost exactly the order the user will want to know it. Material is concisely presented; points are explained from the ground up; and discussion of "qualifying instances," which gets so many manuals off on distracting, confusing tangents, is largely postponed. (The manual has more appendices than chapters, and they occupy almost as many pages.) The twenty-page appendix of error messages is subdivided by type, and each follows a clear "error/cause/solution" format (although messages are perhaps too heavily weighted toward problems in printing). The first four chapters have users doing practice exercises to reinforce what they are learning. The fifth chapter, an "Editor Reference Guide," starts with a paragraph on "General Principles" and follows with very clear and concise explanations of functions, grouped by major area rather than arranged alphabetically. Although the manual cautions that this reference guide is only meant to be used as an adjunct to the on-screen tutorial, most users will probably find it is a much more efficient and speedier way of reviewing the command structure than the rather mediocre and unimaginative tutorial.

The intent of Volkswriter in terms of the word processing market is signaled by the fact that it includes a sub-program, indicatively titled "WSTOVX," just for the conversion of WordStar files to Volkswriter files. Obviously, Volkswriter's makers are after "big game." What they are offering is a system that is as conceptually clear, if not as powerful, as WordStar--the major difference being that Volkswriter emphasizes simple commands within a decidedly eclectic schema, whereas WordStar offers complex commands within a very orderly schema. Certainly, for a number of advanced student writers--Volkswriter would be an adequate, but hardly compelling, choice. Beginners--and not a few academic writers--might well award the palm to Volkswriter. For anyone working in French, Spanish, Italian, or German, there would not even be a contest; for now, Volkswriter is the program for multilingual word processing. Still, anyone drawn to Volkswriter should check its features very carefully: its design reflects some very definite "second-generation" notions of what a word processor should and should not do, and those notions might not necessarily coincide with the potential buyer's.

OFFICEWRITER

Distributor: Office Solutions, Inc.
5708 Odana Road
Madison, WI 53719
608-274-5047

Cost: $325

As its name and the name of its manufacturer clearly imply, this is another of those word processing programs oriented toward the workplace. One of OfficeWriter's strongest selling points is that it has a document merge capacity built into it, so a separate program need not be bought for producing

customized letters, printing envelopes, and creating and filling in forms, all important functions in business offices.

According to its makers, OfficeWriter is specifically not modeled on other microcomputer word processing programs, "most...developed with an eye toward Wordstar," but instead "was inspired by the dedicated word processors such as Wang, Lanier, and the IBM Displaywriter." In this case, as in many, inspiration is not coequivalent with actualization: OfficeWriter is still very much a microcomputer word processing program, though by no means a bad one.

A number of its commands are single keystrokes: like F1 for centering, F5 for underlining, F6 for boldfacing. While these commands do not take effect on the screen, their symbols--respectively, a pair of diamonds, double exclamation points, and raised squares--are easy to spot and do not distract much from the text. The end-of-paragraph symbol, a solid triangle pointing left, is also very "readable," and properly definitive. Although not screen-oriented, OfficeWriter is quite screen-readable.

Many keys are logically employed: e.g., Ins starts insertions and Del starts deletions. Whenever Ins is keyed, OfficeWriter moves whatever text follows its entry point down to the last line of the screen display, thereby making a large opening for additional text and obviating the potential distraction of the following text constantly reshaping itself as each new character is typed. Deletion is quickly accomplished by pressing the Del key, moving the cursor over the text to be deleted (which is highlighted), and striking "GO" (the large key, with + on it, found at the extreme right of the numeric keypad). If a mistake is made while inserting or deleting, the process can be cancelled with F10.

OfficeWriter goes many of the more expensive programs two better when it comes to line spacing. First, when document spacing is to be consistent throughout, single- or double-spacing can simply be indicated on the print menu. (Spacing is always single on the screen.) Second, when irregular spacing is required, it can be accomplished simply through the use of multiple "format lines" (what some programs call "rulers")--and spacing can be half-line or quarter-line, as well as single, double, or triple.

A number of OfficeWriter's design features are slightly different from those on other programs. The cursor in OfficeWriter, for instance, is a blinking underline, and the "GO" key is struck to "enter" commands, with the "Return/Enter" key used only to create new paragraphs. F10, not Esc (redefined as the "Help" key), is the command to interrupt most functions. To emphasize word separation, OfficeWriter places a short line (2/5's the length of a hyphen) in the middle of the spaces between words. The program eschews the Ctrl key entirely, using Alt in its stead--for operations like large cursor movements (Alt + left and right arrow keys, Home, End, and other keys) and subscript/superscript (Alt + up and down arrows). In addition to the cursor, the program also has an end-of-document marker (a solid rectangle). Besides letting the user choose character spacing, justification on/off, page pause, number of copies, lines per page, and other print functions, the print menu allows the user to set page numbering to start with any number (something difficult to effect in some programs and impossible in others).

The business orientation of OfficeWriter is exemplified by the "Document Summary" screen that appears first whenever a document is created or edited. Under the document's file name there are seven lines of information that can be filled in: description, author, operator, comments, date created, date edited, and size (in characters). Obviously, only the most regimented of personal users will bother filling in this information all the time, but these document summaries could be very helpful for keeping track of office workflow, particularly in an office where two or more persons often work on the same document. (Document summaries can be printed out with a document.) Another function likely to be more important in a business situation--although it does have applications for the classroom and for some kinds of academic writing--is "SuperMove"/"SuperCopy," which operates just like "Move/Copy" except that it moves (copies) blocks from one file to another.

A useful, and rarely found, capability that OfficeWriter has is the "Go-To-Page" function, which allows the cursor to be moved to any page in a document file. Thus, it is much easier than in most programs to get anywhere in a long file in a hurry. (Many programs require slow movement from the front or back of the file a screen or page at a time, or presetting of markers in the text so the program can be directed to go to them.) While OfficeWriter does not allow for on-page footnoting, it is very flexible with headers and footers, which can be three lines long and up to 176 characters. Actually, footers could be used as a substitute for on-page footnoting if no footnote is more than three lines, which is often the case when footnotes are just citations for quoted and referenced material. Because headers and footers are created on a "page" of their own, it is possible to place material anywhere one wants in relation to right and left margins. It can also automatically be set to print from the left or right margin on alternate pages (to resemble alternating headers/footers in books and journals).

The program can do horizontal scrolling, up to 156 characters, a very important asset when editing any document that has lines that exceed the eighty characters visible on a line of the display. Hard ("binding") spaces are entered with Alt= and permanent page breaks with Alt F2. Decimal or letter columns can easily be moved, duplicated, or removed with a series of Alt commands.

OfficeWriter is not a program without design weaknesses. For example, the function keys are often used logically in pairs (F3 is "Search," Shift F3 its extension, "Replace") and even in fours (F7 is "Move," F8 "Copy," with the Shift versions being "SuperMove" and "SuperCopy")--but as a group of twenty operations, the function keys seem rather miscellaneous and so somewhat difficult to keep straight. F4 calls up the format line; Shift F5 marks text for double-width printing; F10 is for exiting to the main menu; F2 creates page breaks. In short, the overall arrangement of commands in the program is not nearly as mnemonic as it could be--and so recalling one command does not always "lead" you to the related one you are looking for.

OfficeWriter's help screens are not very helpful, since they are densely packed and not especially easy to read: they consist almost entirely of text, avoiding lines, boxes, arrows, oversized characters, and other visual aids that might have been employed to highlight material and make it more graspable. And in the early going, a user is likely to need help fairly often, particularly because the destructive backspace key is not used in

OfficeWriter, and so the novice may have to keep checking to see how typos are corrected. To be sure, the regular deletion method is clumsy, involving four keystrokes to replace one wrong character: backspace, Del, GO, and the correct character. A simpler way is backspacing, then hitting the desired character, which will replace the mistaken one, since the "overstrike mode" is the program default. However, this method will only work when one character (or space) is replacing another character (or space), and not when the user is trying to eliminate a letter that was accidentally typed twice. In addition, working with overstriking can be confusing for someone still trying to master the concepts and processes of insertion and deletion.

Another problem with OfficeWriter, though by no means an insurmountable one, is its on-screen tutorial (which comes on a separate disk). It covers only the basic program functions and does so in a way not very different from the manual. Like the help screens, the tutorial screens are dense with text and ignore most of the many ways material might have been visually highlighted. The practice tasks it demands do little to acquaint the user with the basics of the program; one is told to scroll through the text, give a "Go-To-Page" command, and return to the starting point ("Did you have a nice trip?"), but not to create and work with any sustained piece of practice text. Whatever text is created is not permanently recorded, making the tutorial "clean" for the next user, but all the less substantial for the present one.

Fortunately, OfficeWriter's manual compensates for most of the weaknesses in the tutorial--to such an extent that a user might do just as well to dispense with the tutorial altogether. In fact, Chapter Two, which is meant to support the tutorial, is much better than the tutorial. It contains a number of diagrams of the keyboard layout, with the specific special function keys then being discussed highlighted in gray. These keys and their functions appear in an easy-to-read box chart beneath the keyboard diagrams. Whenever sample text is shown as it would appear on screen, it is in a highly contrastive bright green in the manual.

For the most part, the manual is logically organized, for example, presenting the print menu screen, then discussing each of its items in the order in which they appear on it. Quite helpful is the alphabetically arranged reference chapter (7), where each referenced command (e.g., "Boldface," "End Merge," "Go to Left") is given at least a page of its own. Each entry starts with the command's purpose and the primary key used to accomplish its task. If other keys are sometimes involved in the operation, they are listed as "Active Keys." Each page concludes with one or more "Remarks" and, in some cases, an example (showing, for instance, the difference between a command as it appears on screen and in the printout). The manual's sixteen-page appendix of clearly explained error messages would be much more useful, though, if possible corrective actions were also included. Another appendix--listing the character spacings, line spacings, and special functions that twenty-five OfficeWriter-compatible printers will support--is itself a very good basic guide to printers.

OfficeWriter is a quality product; for a very small example, the molded plastic template it provides to fit over the function keys will last much longer than any users will need it. In the best sense, OfficeWriter can be said to be a steady, dependable, unspectacular word processing program. For use in a classroom setting, the instructor would probably want to work out a

selection and rational schematization of its rather arbitrarily arranged commands. For personal use, faculty might discover that OfficeWriter meets all their varied editing, formatting, and merging requirements--and so can save them as much as the $400 it would cost for a more expensive word processing program and its allied mailmerge program. Even for those who prefer screen-oriented programs, OfficeWriter is worth investigating.

EASYWRITER

Distributor: IBM Corporation
P.O. Box 1328
Boca Raton, FL 33432
800-447-4700

Cost: $175

EasyWriter has a bad reputation--in large part deserved--but it is not an impossible program to work with, a fact which should prove comforting for faculty at institutions which rely almost exclusively on IBM hardware and software. (Cognizant of widespread dissatisfaction with EasyWriter, IBM has taken to offering three other word processing programs on its standard software list: PeachText, VisiWord, and WordStar. Of course, any of the programs analyzed in this monograph will run on an IBM personal computer.)

Computer mavens seem to take IBM's selection of EasyWriter as its word processing program almost as a personal affront. As one of them put it:

> It's hard to understand why IBM would choose EasyWriter for their Personal Computer when so many better word-processing programs are available. EasyWriter is far and away the worst program I evaluated--inconvenient, poorly thought out, and full of bugs. That the largest computer company in the world should offer it as the first (and, for a time, the only) word-processing program on its personal computer is, in a word, disgraceful. It just goes to show you--you can't trust a name. (Naiman 115)

By now, as this evaluator went on to anticipate, most, if not all, of the bugs have been worked out of the system (I tested version 1.15). Still, the clumsinesses, inconveniences, and distractions remain. (EasyWriter's designer, Information Unlimited Software, found the very best way to get rid of such problems: after selling EasyWriter to IBM, IUS developed a new, much better program--EasyWriter II [evaluated below]--written along very different lines and in a more sophisticated computer language.)

Although it is by no means the oldest microcomputer word processing program around, EasyWriter is very much a first-generation program, one which has a number of ungainly and unwieldy operations that even the more inexpensive newer programs have managed to avoid as a matter of course. It

has only eighty-eight commands, but still relies on a miscellany of ways to enact them: via menus, control commands, and embedded ("imbedded") commands. Its editing power and on-screen formatting are weak despite this mixture of command structures.

EasyWriter does not always take advantage, or even make logical use, of the IBM keyboard. For example, striking Home will move you to the top of the screen, but the program uses its complementary key, End, in conjunction with Ctrl to delete the rest of a line. The commands on the main menu are not discrete: one must "Get" a file before one can "Edit" it. The first time, a user "Saves" a file and each subsequent time "Revises" it (i.e., saves a revised version). Justification is one of a number of functions that includes extra steps: after getting to the Additional Commands Menu, you hit J, Return, then A (Align), then J, Return, and F4 to get back to the text. Underlining is even more of a morass, taking about ten steps, including two embedded line spacing commands which put the underlining on a line of its own over the word(s) and direct the program to do "SPACE0" so that line comes out on the same line as the text to be underlined. (Imagine trying to explain this to a student who off-handedly asks how to underline a title in an essay.)

Some indication of the scattered organization of EasyWriter's command sequence may be gleaned from the very mixed set of commands it assigns to the ten function keys: Help On/Off, Print file(s), Insert Line, Addn Commands, Delete Word, Undelete, Stop Print, Block Marker, Align Marker, File System. (There are also ten other mixed commands on this "Editor Help Menu.") Some sequences are not particularly easy to get down--for myself, I instinctively tried to go back to the Editor Help Menu from the Additional Commands Menu by F1 (which is how you get there from the text), but this just put the Additional Commands Menu back up on the screen; in this, but not in too many other situations, a toggle function is employed, F4 getting you to and from the Additional Command Menu.

Keys are not always employed as you would expect in EasyWriter. Although Ins is involved, for what exact reason I am not quite sure, in the complicated block move (or copy) operation, it is not Ins which actually inserts the designated block, but F3. Such things can be learned with time, but the cumulative effect of having to learn a dozen complicated and not particularly logical sequences makes EasyWriter a much slower system than most on which to build up to your normal writing speed. For students who need to learn a system quickly so they can get on with the writing that is the real work of their course(s), EasyWriter would be a most frustrating system on which to have to learn.

The manual has a twenty-two page appendix that just lists the changes between version 1.0 and 1.1 of EasyWriter, a supplement which rather belies the quality implied in the program's name. And when the program is "easy," it is often so because a function is lacking and cannot be performed. EasyWriter lacks soft hyphenation, a delete buffer, decimal tabs, scrolling (by line), boldfacing (it does have double-strike), and, for those who want them, automatic backup files--to say nothing of its missing more advanced features like split screens, double-column printing, and footnoting. A potentially more serious limitation is the amount of text it can move or copy at once: 4,000 characters (a little over a page, and only half that if the computer has but 64K). For any writer who works by moving around large sections of text,

this procedure would necessitate so many small moves that the usefulness of the process would be very much diminished. Nor does the EasyWriter screen provide much information about where you are in a file to make moving text any easier. Also, moving/copying is another series of functions that starts, for some reason, with Ins, after which you go to the Ctrl commands that actually perform the operation. Last, but certainly not least, among EasyWriter's drawbacks is the fact that it is easier to lose a file than it should be. (To be literal: losing a file should never be "easy.")

While these deficits are probably enough to turn most potential users away from EasyWriter, the program is not without some definite assets. Its print commands, for instance, include "Print to Screen" so the user can see just what the text will look like when printed. (The PrtSc command also works in EasyWriter.) And for printing to the screen, as well as for regular printing, there are four single-keystroke commands that give the user much more (and often needed) control over the process than most programs: Slow printer, Resume normal speed, Pause printer, Stop printer. EasyWriter allows for the embedding of user-designated commands that can take advantage of the special attributes of printers. It has a command to print a file from the cursor forward, which occasionally can be a helpful function, and it will, upon a simple direction, print multiple copies.

Saving a file and returning to it is a quick, efficient process in EasyWriter. (Here, as at several other important junctures, the program asks "Are You Sure?" before replacing old material with new.) Commonly used sets of embedded commands can be saved to a "header" file and inserted at the top of new files with a few keystrokes. The program also has a RECONFIG utility, which will permanently readjust default values (on line spacing, margins, lines per page, etc.) if the program's ordinary defaults are not to the user's liking. On the Addn Commands Menu there is a command that gives a word count of the document currently in memory (a feature most students love, even if it is not always as useful as it may seem).

Some of EasyWriter's design features do make the program easier to work with rather than harder. Its end-of-paragraph symbol, an eighth note that appears in the column where it is put (and not in the margin), is definitive and easy to spot when it comes to making alterations. F5 (delete a word) and F6 (undelete) are useful, paired single-keystroke commands. (Unfortunately, undelete does not work for more than a word.) F3 inserts a blank line, so longer insertions are easy to see on the screen. EasyWriter can append files to one another--at least 27 and as many, if one can imagine doing this, as 124. It can also link files, so they can be searched across as one unit. Large deletions can be done by marked blocks. EasyWriter's help menus are more helpful than not (and more helpful than most).

The EasyWriter manual does not, except in a few cases, compound the difficulties of operating the program by making explanations of operations equally difficult and obscure. A looseleaf binder, with high-quality print (screen text well highlighted in an easy-to-read green) and thick paper, the manual is divided into three sections. The first explains, quite clearly, how to set the program up on the IBM system. The second is a series of five "lessons" which, despite their sometimes arbitrary and directive character, do a fair job of getting the user working on the system. The reference section, the longest of the three, is clearly written but necessarily verbose, since

explanations of so many functions must be long and complex. The index is excellent and full. The eleven pages of error messages relate chiefly to problems of machine operation and file management. Another appendix, on "Tips and Troubleshooting," may betray the program's overall weaknesses, but at least it helps the user to compensate for them. The foldout reference card usefully collects the rather miscellaneous major commands (and menus) together for quick referral. Although some reviewers have scored EasyWriter's manual for being incomplete, I have not found it so. Overall, EasyWriter's documentation is rather better than average--would that the program were as good.

Any teacher employing EasyWriter with a class would have to rethink its command structure and devise a direction sheet--or perhaps a series of them, increasing in complexity--to make the program appear less arbitrary and chaotic, more organized and workable. Any new user would do well, in the face of early frustration, to remember that certain commands will soon become easier and quicker. For example, he or she will soon be able to remember the centering command, thereby avoiding having to go "through" the Help Menu to reach it, meaning three keystrokes (F1-F4-C) and a scanning of the screen can be reduced to two keystrokes (F4-C) without a searching of the screen. Withal, EasyWriter is a program, as subsequent developments in microcomputer word processing programs demonstrate, where many functions are more complicated than they need to be--and it is a program whose complexity of operation does not compensate the user with many special features. EasyWriter's price has gone down from $350 to $175, but for most kinds of work, it is possible to get a better program for less and a much better program for not that much more.

SUPERWRITER

Distributor: Sorcim Corporation
2310 Lundy Avenue
San Jose, CA 95131
408-942-1727

Cost: $295

SuperWriter is a clone of WordStar. Not only are its overall logic and organization similar, but a good many of its commands are exactly the same. For instance, it uses the same diamond of six letters--EASDFX--with Ctrl to move the cursor; likewise, it scrolls up with Ctrl W and down with Ctrl Z; and if the delete character-word-line pattern is altered slightly (from Ctrl G-T-Y to Ctrl G-Y-N), it may only be as a means of helping to avoid copyright infringement.

Quite naturally, since SuperWriter is based on a good, reliable program, it is itself a generally good and reliable program--and one which tries to improve on WordStar in some small ways (and sometimes succeeds). It also sells for less than WordStar and includes in the main program two

capabilities, a spelling checker and a mailmerger, that must be bought as peripheral programs with WordStar.

Throughout, SuperWriter has the look of a very full program: a densely packed manual of nearly 300 pages, some help screens ("Answer Screens") that fill up the video display with as many as 40 varied commands, and a foldout "Answer Card" that lists 109 formatting commands (more commands than some intermediate programs, like EasyWriter, have altogether). One indication of SuperWriter's fulsome approach is the very complex tree diagram which is printed on the other side of the "Answer Card" from the formatting commands, and, in a very slightly expanded form, in the opening pages of the manual. It outlines and interrelates all the editing commands, as well as the major steps under the other five choices on the main menu: "Quit," "Print," "Check Spelling," "Disk Directory," and "Utilities." For experienced word processors this diagram will provide a very complete overview of the program and will function as a very efficient guide to its operations, but for the novice or inexperienced word processor, or even for one used to working with simpler programs, this diagram will be daunting and discouraging.

SuperWriter's documentation does not entirely ignore this latter group--in fact, it goes, if only briefly, to the other extreme for them. Included in the program package is the pamphlet "10 Minutes to SuperWriter: Create your first business letter in 10 minutes." This "walk through" of the program, during which the user creates a short business letter, is clear enough, if awfully and incorrectly written ("Alright, now start typing. Yep, just like that")--but it is, on balance, too simple. The novice might come to understand a few things about SuperWriter by using "10 Minutes," but essentially its directions create the letter for the user--after which he or she is up against the stone wall of the manual. Like WordStar, SuperWriter is a program that needs to be learned slowly if it is to be mastered--but its producers seem to have devoted most of their effort to winning over converts from WordStar (and other programs of that caliber) rather than to making SuperWriter accessible and useful for users newly come to word processing. SuperWriter's designers make a cogent observation, if only in passing, in "10 Minutes": "Most of the editing you will need to do is just a matter of deleting one thing and inserting something else." Unfortunately, they do not keep this observation in mind in the opening chapters of the manual, but instead overload the user with information on dozens of functions.

As one would expect, SuperWriter lacks some of the features that WordStar also noticeably lacks, from minor omissions (the capacity to move to the middle of a screen) to potential major deficits for certain kinds of writing (like the capacity to do on-page footnotes). It follows WordStar in providing the "safety" feature of automatic backup files and in not providing a delete buffer to complement this feature on a smaller scale. Like WordStar it cannot do split screens, although its "library" menu compensates for this lack in some degree. And like WordStar it does not enact some commands, such as underlining and boldfacing, on the screen, thereby compromising its screen orientation a bit.

Some of the differences in commands may prove distracting to those who have used WordStar (a situation that obviously holds true in reverse). For instance, SuperWriter uses the backslash [\] rather than the dot to signal embedded commands. Its superscript and subscript commands-- <..> and

>...< --might be clearer than WordStar's on the screen, but they are not as easy to get straight as WordStar's superscript and subscript commands (Ctrl PT and Ctrl PV) since they are not true toggles but alternations. Whereas WordStar makes the direction to discard a file without recording any of the changes made as part of a series of "save" alternatives (Ctrl KQ), SuperWriter gives it an independent stature (the "Zap" command). Unlike WordStar, SuperWriter asks for some information about the document that will be created (number, author, comments, etc.) before giving the "page" on which to start recording it--a feature that might be useful in an office setting or distracting in a personal one. SuperWriter does allow a quick exit to its allied SuperCalc program if the user has it.

SuperWriter improves on a number of small design features in WordStar. It offers, for example, a righthand set of four cursor movements--Ctrl H,J,K,L duplicating Ctrl A,S,D,F--that users may find speeds cursor movement. Some commands are more mnemonic (Ctrl PU rather than Ctrl PS is the underline toggle), and some menus (particularly the main menu, which spells out options) are clearer. Its line spacing is more flexible in that it allows for half-line spacing, and the program will do proportional spacing for printers that support this feature. It has a "page insertion" feature for longer insertions (the text after the cursor is moved all the way to the bottom of the screen), which obviates the putting in and taking out of several blank lines, an action many WordStar users perform to make longer insertions clear. Unlike WordStar, SuperWriter easily permits global changes in line width, tab stops, etc. It also has a "Library" utility that will display a second document without interrupting the editing of the first. (Changes in the "Library" document, however, are not possible--a fact which makes this feature less useful than "windows" where the user can switch from one file to another.)

Certain of SuperWriter's design features seem less logical than WordStar's, such as using Esc S for the save command. (Here and elsewhere SuperWriter uses Esc, which many programs, including WordStar, use as the key for cancelling functions or as the key that starts major preservations or shifts of text.) Its whole move/copy sequence is more complicated and less clear than WordStar's and risks some confusions with embedded commands by making the slash [/] the symbol used to mark blocks. Also, it is impossible to create a backup file that is exactly the same as the main file--a matter of simply saving a file twice in WordStar--without taking some steps, like adding a character, then deleting it, to indicate to the program that some sort of change has been made. (It is often a useful standard procedure to make the main and backup files identical at the end of a session, so two copies of the most recent version of a document are preserved.)

I also found two minor distractions that had nothing to do "comparatively" with WordStar: 1) when I returned to the text from a menu, I would find an end-of-paragraph symbol that was not there before (perhaps just a flaw on this program diskette?); 2) a whole series of alternate function key commands is misrepresented on the menu: "shifted" function keys should be represented, e.g., as "sF8" and not "F8s," since hitting the function key first will give you its unshifted function.

Most significant is the limitation of SuperWriter's spelling checker. The problem is not that it is not as good as the version of SpellStar (since

revised and much expanded as CorrectStar) upon which it is based, but that it is only as good, and that is not half good enough. Both are based on 20,000 word dictionaries, and this number of entries is simply too small to allow variant endings on all nouns and verbs to be included, to say nothing of only slightly uncommon words. (Using the early version of SpellStar, one of my students was astounded that it kept marking "baseball" as a possible misspelling.) Thus, SuperWriter's spelling checker only produces a primitive kind of proofreading for spelling errors. Still--and especially since it is "free"--it might serve to help a very poor speller catch some errors, and it might help reinforce correct spellings by prompting the user to doublecheck a number of words that are right. However, any user who wants to do a more thorough, but inevitably incomplete, check of spelling would have to purchase one of a number of independent programs designed for this purpose.

SuperWriter is a program with a plentitude of features--and one whose manual reflects the best and worst of such a full program. It has some good, enlightening diagrams and any number of illustrated screens, which are conveniently listed following the table of contents. It could, nonetheless, use even more screens, especially ones showing how documents look on the video display. The manual has a good, full glossary, a list of its many (38) custom default settings, even an appendix with standard proofreader's marks (which can't hurt). The print--the whole manual "was written, edited and revised using the SuperWriter program"--is distinct if a bit light for some readers, but it does provide a sharp contrast for the boldfaced (and darker) text that is scattered throughout. At times--such as the section on moving and copying--the manual confuses the reader by having the text used to illustrate operations also be about those operations. The biggest fault of the manual, beyond its lack of concern for the novice, is its wholly inadequate index. For instance, on the "Answer Screen" that provides a list of the major edit keys is the designation "ALT FUNCTION KEYS...Ctrl-Q". The user trying to find out from the index what this means will not find "ALT" listed at all--and will find nothing under "Function" or "Keys" that is illuminating. Many other terms will be equally perplexing for the user who relies on the index to search them out.

One further problem is that, as far as I know, there are not, as there are for WordStar and a few other complex programs with forbidding manuals, simplified commercial introductions and guides to the program. With SuperWriter one must face up to the manual.

The user who works through the manual, though, will have a good, generally powerful program with which to write, edit, and format--a program which will take some time to learn, but which operates smoothly once the novice gets the hang of its quite logical operation. Whether it is fully the equal of WordStar is difficult to decide--but certainly the SuperWriter user will be able to write in the insert mode, link files and create a table of contents, and create large spaces in a document without having to see all of the space on the screen and work in dozens of other ways that signify word processing at a high level. Because it is a program in the $300 range which is modeled on one in the $500 range, SuperWriter is, as its makers doubtless planned it to be, a "bargain version."

BEST COPY AVAILABLE

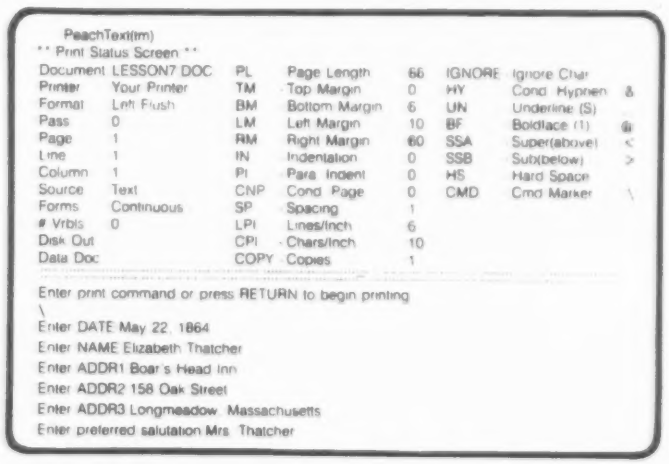
PEACHTEXT

Distributor: Peachtree Software
Management Science America, Inc.
3445 Peachtree Road, N.E., 8th Floor
Atlanta, GA 30326

Cost: $249

PeachText, formerly Magic Wand, has recently been discounted (from $400) into the intermediate category, where it is now a definite bargain for users who need a program with powerful and varied print capabilities. PeachText is far from being the easiest program with which to work on the screen display--although its thorough documentation does help in this regard, but it will support a number of print options (e.g., true proportional spacing and nine intensities of boldfacing) that make it one logical choice in situations where documents must approximate the regularity, visual variety, and orderliness of published texts. No other program anticipates and allows for such a large number of variations in printed format, although several more significant print options, like on-page footnoting and "print pause" (so alternate typefaces might be installed), are not included.

PeachText is clearly not a program designed for casual users, or even for regular users with relatively straightforward and unvaried word processing requirements. Except in the rare cases where students need to produce complexly formatted documents, it would hardly be a logical or effective choice for student work. Learning PeachText takes a fair amount of time; becoming comfortable with it takes many more hours still. Consider, for instance, how long it would take to come to understand the following screen, a simple example of what one encounters in doing a document with "variables":



The whole Peachtree line of software seems oriented toward the "serious" word processor, typically a business user. Although PeachText is available

BEST COPY AVAILABLE

separately, the company is obviously more interested in selling it as part of its PeachText 5000 package, which it alternately types as a "Personal Productivity System" and an "Office Productivity System," the alternation being a possible suggestion that the personal user would do well to organize his or her work along business lines. Besides the two-disk word processing program, the six-disk PeachText 5000 package includes the Random House Electronic Thesaurus, a Spelling Proofreader, a List Manager (a "control center" for the entire system), and PeachCalc Electronic Spreadsheet. The whole is an impressive package--Peachtree is very good at packaging its wares--but it clearly contains more programs than most academics, save the most regimented and record-oriented, will need.

The PeachText word processor itself is a complex, menu-driven program; through its heavy reliance on embedded commands (indicated via a backslash), it is able to avoid Alt key commands entirely and to use Ctrl key commands sparingly. Its menus tend to offer many choices at once, but those choices are typically well laid out and comprehensive. For instance, the main menu offers fourteen choices (all but "?" two letter commands); eight "Document Commands" (edit, print, copy, delete or rename a file, display directory, get help, exit the program), two "Disk Commands" (swap disks, change default), and four "Special Commands" to get to allied programs (the spelling proofreader, list manager, PeachCalc, and Telecommunications, a program not included in PeachText 5000). Together, these commands allow the user to begin (and end) any word processing activity or any related computer function. Similarly, the seven help screens--two alone on print commands--are very full and concise; what they lack in instructional clarity they make up in informational completeness.

PeachText's command structure is decidedly quirky, a characteristic that makes the program just that much harder to learn. Some commands are designed on a mnemonic basis: e.g., BM, BC, BD, and BK are the commands for moving, copying, and deleting a block of text and for "killing" block markers. Other sets seem rigidly logical: e.g., BF1-9 sets the intensity degree for boldfacing and BFØ turns boldfacing off. (Yet, "Edit" on the main menu is the starting command for both the creation of files and the revision of them.) Still other sets seem to scatter individual commands over the keyboard in rather random fashion: e.g., single characters are deleted via either the backspace key or Del (which eliminates the character under the cursor); words are removed with F4; and F9-F9 deletes a line. (F9 needs only be pressed once for each subsequent line deletion.)

As with the line deletion command, PeachText sometimes illustrates a mild repetition compulsion. Search is done by typing F6 and the unit to be searched; searching and replacing calls for a second F6 set, with the replacement unit typed in. Striking Home once moves the cursor to the beginning of a line, twice to the top of the screen, and thrice to the bottom of the screen (thereby ignoring a logical use for the End key, which is paired with the Home key on the IBM keyboard). The two methods of entering insertions typify PeachText's somewhat arbitrary--and hence difficult to recall--way with repetitions in commands. Small insertions are done by pressing Ins to start and any other control key to end, whereas large insertions start and end with F8 (which, it should be noted, very helpfully clears a large space by pushing the text following the cursor to the bottom

line of the screen display and eliminates that line if the insertion reaches to the bottom of the screen).

A good deal about PeachText--most particularly its essential postulation of an office setting--is indicated by its mechanism for "variables." Up to 128 of these, each up to 55 characters long, can be created for a document. Together with the procedures for creating and inserting data files, variables make for virtually a full-fledged mailmerging sub-system, although this feature is never identified as such in the documentation. Clearly, PeachText's makers saw the creation of form letters and like documents as such a routine part of the program user's work that the features for doing such were integrated fully into the most normal program operations.

Some of PeachText's organization around the print phase may be indicated by the fact that "saving" does not, as it does in most programs, refer to the permanent recording of a document on a disk, but only to a temporary preservation of it if an edit is interrupted. One "saves" in the more usual sense by "ending" an edit, after which one can "print": on the printer, on the screen, or on a disk. It is this orientation toward the print activity that makes PeachText a powerful program--and a difficult one to master. An experienced user can direct the printer (if it is capable) to justify text by inserting small spaces between the letters rather than whole extra spaces between words, to print an entire text or any designated parts of it in line spacings ranging from one to six in half-line increments, to force a page break wherever wanted, to shift back and forth between solid and broken underscoring, to print with the right margin flush and the left ragged, to put in headers and footers of any length, to print any number of copies, to make pages up to 255 lines long, to insert conditional hyphens, to "print in background" (i.e., spool), and to obey commands inserted for a specialized printer when the document is to be printed on a non-specialized printer. Some users will undoubtedly find PeachText's wealth of print options a distraction, and others might be tempted by such possibilities to fiddle with documents to unproductive lengths--but PeachText would do very well in a situation where the user produces a steady stream of notices, memos, reports, and other documents for "in-house" use. PeachText allows for both visual/format variety and an approximation of the look of "published" texts--and these are characteristics likely to attract reader attention and increase reader concentration, especially when readers encounter a fair number of documents each day. Since formats can be stored and "pulled out" as needed, the creation of highly specialized layouts will, after a time, be quickly accomplished.

Not all of PeachText's assets relate directly to printing. For instance, the program automatically creates backup files (the penultimate version) of a document as a safety feature; it allows for the insertion of unprinted notes as reminders to the file's creator or explanations to its on-screen readers; and it will easily display the available buffer space, so as to make the user aware of the possibility of creating a document that might be too long to fit. (PeachText will create documents up to the length permitted by the system being used, but the documentation suggests the user limit files to the size of the computer's active memory--something that is usually easier to do in an office setting than in a scholarly one.) PeachText's kerning capability, which moves the print head any designated number of small increments left or right is not only useful for some special print effects, but also can be set

to zero so accent marks can be placed over letters to form foreign language characters (which might save some foreign language scholars the expense of a specialized printer, provided they did not mind the extra time and effort using this method would entail).

Like its overall design, PeachText's documentation presumes a user who will invest substantial time in return for learning a complex, many-facted program. Although PeachText 5000's documentation is the most thorough of those considered in this monograph, it is, rather oddly, without a program manual per se. Instead, there is a very complete "Lesson Plan," which takes the user through all the program functions in PeachText and in other programs in the 5000 package. There is also a large reference guide, which also covers all the programs in the 5000 package. (Presumably, if you buy just the word processor, you get shorter versions of these same texts, with the latter sections on the allied programs excluded.) These are very professional looking publications: spiral bound, with attractive color covers, high-quality graphics, copious subheadings in green (in a column of their own on the lefthand side of the page), and many sample screens (clearly blocked, with screen text also in green). Despite the size of these publications, explanations in them are almost always clear and concise. (PeachText simply has a great deal it must explain.) The indices are thorough and refer not just to pages where the information can be found, but indicate to which program(s) the entry refers. In addition, Peachtree makes available, at extra cost, an on-screen tutorial, created by ATI (American Training Institute), for the whole 5000 system. Still, users who might prefer a manual which concentrates on giving a section-by-section overview of the program will find themselves shuffling between as many as four documents--the lesson plan, reference guide, on-screen tutorial, and the user's handbook that accompanies the tutorial to get such an overview.

The lesson plan is a good, fulsome one, with explicit exercises to test knowledge of all the many functions it teaches. Its work with an on-screen sample document, which is carried through several chapters and updated and rearranged, is effective, as are the sample display screens. The slowness of the lesson plan is balanced by its thoroughness; anyone who takes the many hours to go through all of it carefully will be able to use PeachText on a sophisticated level when done. (It might, however, have been more useful to construct the first few chapters as the equivalent of a "Quick Start" course, so users with simpler word processing requirements or those in a hurry could get to work sooner.) A few of the lesson plan's exercises--particularly the one which presents a wordy and flat modern version of the Gettysburg Address for revision back to its original form--seem banal when they mean to be clever. The tone can take on, like so much in the realm of word processing documentation, an unpleasant and unjustified complacency, but all except the excessively finicky will be able to get through the lessons efficiently and profitably.

Only users who have a strong prediliction for being instructed and congratulated ("You are doing very well!") by computer will want to bother with the ATI tutorial, which advertises its ability to help "you learn the most commonly used PeachText operations in less than an hour." Since PeachText is a complex system, those "most commonly used operations" do not amount to very many. The ATI tutorial is not especially sophisticated graphically or instructionally, but it is as clear as the rest of the

PeachText documentation--so the user is not likely to lose anything (or any time) by learning the program through it. Its most effective feature is that it regularly allows for the review of material; its least effective is that it sometimes directs you back to printed matter ("Use the User's Handbook").

The reference guide is probably the best thing of its kind on the market. For experienced word processors it alone can function as almost a full explanation of how things work with this program. For those who become experienced with PeachText, it is an excellent means of learning and incorporating into one's "repertoire" more and more specialized functions.

For all of its sophistication and print power, PeachText has some surprising omissions in its design, several of which immediately rule it out as a program to use with students, except those already quite familiar with word processing. There is no status line indicating page, line, or column--the text is just up there on the screen. Text does not appear justified on the screen, nor is it centered. (Even a number of print-oriented programs do enact the "center" command on the screen.) Neither does boldfacing, underlining, superscripting, or subscripting of text appear on the display. Instead, the screen is full of embedded commands, which undoubtedly will prove distracting to the novice who is trying to concentrate on what he or she is "writing."

As for editing, a major PeachText weakness is that it provides no easy method for intermediate cursor movements; e.g., going 45 characters to the right on the line must be done one character at a time using the right arrow key. It is quicker to move twenty lines to the top of the screen than ten spaces right or left on a line. When one scrolls up or down, the screen, very distractingly, rewrites the screen with every line. This Ins toggle cannot, for small changes near each other, be left on, since moving the cursor even one space turns it off, thus slowing down the revision processing by making every change an act that must be begun and completed before performing the next one. To put into the printed text certain "recognition characters"--such as @ and & which normally function as part of the command structure, one must first "cancel" their command value, a fairly simple procedure, but one which can cause havoc in the document if it is neglected. For those working with the 5000 package, the 20,000 word spelling proofreader will prove of very limited use. (See remarks under the appraisal of SuperWriter.) The Random House Electronic Thesaurus, however, will be a help for some users, particularly student users.

A few, not very significant, print features are missing in PeachText: doublestrike (nine intensities of boldfacing suffice) and strikeover (rarely necessary, except for those followers of the French literary critic, Jacques Derrida, who often need it to put a word or phrase "under erasure"). For those who want them--chiefly teachers of writing who desire students to write imitations of other texts--there are no windows in PeachText for viewing two documents simultaneously. (PeachCalc, though, does have windows.) Nor does PeachText allow for on-page footnotes--although its ability to create footers (and headers) of any size could be employed as a stopgap measure if on-page footnotes were required and the user could adjust to having to move them around, or even redo them, quite a bit. Similarly, those purchasing only the word processing program would have to make do with the "include" feature for combining files since the combine command is part of the List Manager program.

Its (now) attractive price aside, PeachText would not be the first program choice of very many users. But neither would it be a bad choice for those who will be doing much and varied work on the word processor. Its utility will be in direct proportion to how many of its many features, particularly print features, will eventually be employed. Looking up a function and finding that it can be enacted in six or eight variations will be confusing to the novice and distracting for the user who produces basically the same kind of documents in basically the same, simple formats, but it will be very helpful (and suggestive) to the user who regularly produces documents which can be made the more effective by a wide variety of highlightings, layouts, arrangements, and a very "published" look.

PERFECT WRITER

Distributor: Thorn EMI
Computer Software, Inc.
3187 C Airway Avenue
Costa Mesa, CA 92626
714-751-3778

Cost: $349

Although far from being the most expensive microcomputer word processing program on the market, Perfect Writer may, in a general sense, be the most complete one. (Of course, "completeness" is a relative concept, dependent on just what combination of common and uncommon features an individual user requires.) Its partisans, of which there are many, argue that Perfect Writer provides features (like split screens and full mailmerging) and functions (like double-column printing and on-page footnoting) that WordStar and other more expensive competitors do not--or, alternately, that it does a better job of providing such features than programs like Spellbinder or WordPerfect. Its detractors argue that Perfect Writer is rather illogically organized and hard to learn, has difficult documentation, and is not sufficiently screen-oriented. As one of its strong supporters concedes, Perfect Writer "is the most well-thought-out word processing program. It is sad that it does not perform the best" (Gonzales 89). For once, the word processing truth does not lie in the middle: most users will probably come to like Perfect Writer very much or learn to hate it in equal measure. In either case the deciding factors are likely to be whether users can eventually gain some ease with its varied and extensive command structure or whether they can learn to read the often complex screen display to get an accurate sense of the text they are creating. Some sense of the program's complexity can be gleaned from the elaborate "Menu Tree" printed in the first chapter and in the rear of the manual: it has 28 boxed menus and nearly 200 entries in all. Users able to handle all this will have a program that should meet the requirements almost any writers, particularly any academic writers, are likely to make of word processing.

Perfect Writer, obviously, would not be a suitable choice for many classroom word processing situations, since many students would not need its special capabilities nor benefit from learning so complex a program. While its many capabilities make it attractive for academic writers, any academic contemplating its purchase would be well-advised to spend some time working with it or another heavily print-oriented program (PowerText, PeachText, etc.) before opting for Perfect Writer. This program may be the best ultimate choice for a fair number of academics with complicated word processing requirements--it can be said straight out that Perfect Writer will do more than WordStar and its other serious competitors--but academic writers may first have to become experienced users before they can adequately assess whether Perfect Writer would be "perfect" for them.

Although a print-oriented program, more of Perfect Writer's commands--over 150 in all--relate to editing maneuvers than to printing procedures. For example, the cursor can be moved forward or backward a character or word, to the front or end of a line, sentence, or paragraph, although many operators will find it difficult to adjust to having some of these as menu commands. In addition, Perfect Writer employs Home to move to the top of a document and End to move to its bottom. Not only will PgDn and PgUp scroll a file a screen at a time, but using Ctrl with them allows for scrolling of text in the inactive window (i.e., the one the cursor is not in) when there are two windows on the screen at once. (This feature might seem esoteric, but it could, for instance, be very useful for students trying to find the section of a document in the inactive window which will be the basis for the imitation they are writing in the active one.) To the extent an operator can make full use of the many commands for maneuvering around a document, Perfect Writer is a very efficient program for editing texts.

The present (4th) edition of Perfect Writer goes a fair way toward simplifying the command structure of the first edition. The Alt key is no longer used, and Ctrl is used more sparingly. In their stead, an interlocking series of menus offers choices to the user. The ten function keys now have only two, instead of four, sets of functions (by themselves and with Shift). Although the "Menu Tree" shows Perfect Writer is still a complicated program, the design shift to a thorough menu organization makes for what is certainly a more logical and comprehensible system, particularly when seen on the screen. Gone are bewildering "duplicative" command structures, like Ctrl X-Ctrl F to "Find Document," which often had little relationship to similar commands (e.g., Ctrl X2 used to be the command to open two windows).

Perfect Writer's documentation has improved with its design, although it still remains denser than that of many other programs. It presents a program designed in terms of a number of possible options--like setting an "environment" where format values differ from those of the document "environment" as a whole. What the documentation also shows is that, like other print-oriented programs, Perfect Writer makes much use of embedded commands (signalled by @), which constitute another extensive command pattern to be learned. Although most @ commands resemble embedded commands in other programs and deal with localized print effects, a few, like the embedded command to include another file, signal large-scale operations. Fortunately, Perfect Writer's embedded commands can be used in tandem--this program's great strength is that it will allow you to do almost anything you want--by "nesting" one within the other,

although the procedure requires what may seem like a bewildering command use of parentheses, brackets, and other punctuation to achieve the desired effect.

Perfect Writer is a very "full" program, both in the procedures it contains and in the number of functions it will perform. It will do on-page footnotes, create indices and tables of content, and even mark an in-text reference (to another section of the text), so it can automatically adjust the reference as the editing changes page numbers in a file. Perfect Writer does most mailmerging operations one better, even those which constitute entire separate programs, by allowing material to be typed into a form letter from the console (instead of requiring that all material to be merged be pretyped). Its split screen capacity is extraordinary: up to seven files can be held in its buffer at the same time, so it is very easy to work with a whole series of texts at once, rapidly moving them in and out of the screen "windows." Shifting between windows takes just a few keystrokes. It is also possible to put different parts of the same text in the screen's two windows, a useful feature when trying to summarize "in other words." Moving text between windows is easily done, and the size of a window can be increased if more of one text is needed on screen than of the other. The program's on-screen lesson plan, which has the learner performing tasks in the active window following (and scrolling up) directions in the inactive window, is not only an effective learning device; it is also a first demonstration of how useful this dual-window capability can be. Perfect Writer's split screen feature comes about as close as possible to one advantage enjoyed by those who write by hand or typewrite; i.e., it puts a number of documents at the writer's fingertips for checking, comparing, shifting, and combining, and does so without the bother of a stack of papers.

Other design features of Perfect Writer are less extraordinary but nonetheless noteworthy. It has a delete buffer. It has insert as the default, as well as a single-key command (Ins) that moves text down a line so longer insertions will be clearer. When saving a file, the user is asked for the name under which the file should be saved; giving it a new name allows the earlier unedited version to be kept also. Each menu has a "Quit" option, a useful and clear designation imported from other programs (like VisiWord). And it can be used on the IBM-PC with a mouse, although a "mouse installation program" must always be run first. (Will it, therefore, be available for use on the Macintosh soon?) One particularly helpful Perfect Writer feature--that it could be set in the "save" mode, which automatically saved a file to disk every 256 characters--has been eliminated in the new edition, although saving a file to disk is easy enough (Shift F9) if you remember to do it periodically.

What makes Perfect Writer the harder to learn but the better to use is that its designers put into many of its operations helpful additional choices. For example, its replace mechanism--which works backward with as much ease as forward--includes a command ("ASK ME") to handle situations where the user is not certain about the efficiency of a replacement. The cursor goes to the first occurrence of the word and offers the operator five choices: YES, NO, CANCEL, EVERYONE (do this and all other similar replacements), and SHOW ME (which puts the new word in and asks the operator to confirm or rescind the replacement). It also has commands that will save small amounts and possibly large amounts of time. (The "REPEAT" command repeats the last command given, including those for cursor movements.)

Beyond the sheer number and bewildering variety of commands, Perfect Writer's command design has a few definite flaws. Putting some cursor movements on a menu was probably not a good idea, nor was requiring that the user specifically indicate that @ commands exist in a file before printing the file. (If the user forgets, the @ command lines are printed.) The process of embedding commands for boldfacing, underlining, etc. is as complex as in most other programs requiring embedded commands--and, as already noted, rather confusing when commands must be "nested." The program's status line is insufficient: it will tell you if word wrap is off or on, the name of file, and a few other things about its status, but it does not indicate the position of the cursor within the file. Although Perfect Writer has been turned into a more screen-oriented program, it is still a print-oriented program at base, and so does not keep its user informed about where he or she is in the document as if the electronic document were at all times already a printed piece neatly and definitely divided into pages and lines.

Before users can encounter any of these problems, they must get Perfect Writer working, no simple task. Thirty-five pages of the manual are taken up with instructions for installing Perfect Writer. There are very many steps, with a number of questions to answer while proceding with each, along with many comments and further explanations that require some fair knowledge of microcomputers (and not just of word processing) to understand. Of course, such a complex installation procedure can be as rewarding in the long run as it may be frustrating in the short. For instance, Perfect Writer has internal programs that allow for configuration of it to the equipment and for the additional configuration of function keys: the utility of both types of operations will only be fully apparent after the user masters the basic operation of the program and is ready to appreciate its specialized features and customize its operations further.

The Perfect Writer manual--a bound "quality paperback" book, which annoyingly will not lie flat when opened to the first or last fifty pages--is, like the program it describes, full, dense, and complicated. However, information is much better organized than in earlier versions, with operations now arranged in the most common order a user will need to learn them--although it is a bit odd to have the chapter on "Printing" in the section on "Advanced Editing Features." No other manual so often refers its readers, via footnotes, ahead: it is difficult to explain many aspects of Perfect Writer without reference to other aspects, and the manual does not like to repeat material. The index is wholly inadequate though--and the 25-page "primer" at the front of the manual is a perplexing blend of information and "previews" of material in the manual proper. The writing is fairly intelligent and informative, pitched at those who already have some understanding of word processors (a wise decision, since the program's users are more likely to be those who are "moving up" than brand new ones). The lessons are similarly geared to what might be termed the "lower intermediate" level. Each "pop up" menu will explain its terms if the user presses "?" while it is on the screen.

Although it is not the most difficult of the print-oriented programs to learn and operate, Perfect Writer is still difficult enough to make it less than a perfect choice for most classrooms. Besides, not very many students will need the multiplicity of features that are the reason for the program's complexity. It would, however, be an excellent individual choice for that growing number of students who are becoming thoroughly familiar and

comfortable with computers, and who, therefore, will not need a program that is simple to work with and easy to understand on the screen. For any academics with more than minimal word processing requirements, provided they think it possible they can learn to work with a program that is non-screen-oriented, Perfect Writer should be given careful consideration. Anyone who will be using word processing in conjunction with publishing cannot afford not to give Perfect Writer a trial.

Because it is written in the powerful "C" language, Perfect Writer can be used with 16-bit and even 32-bit microcomputers, insuring its longevity as microcomputers advance technologically. Its designers boast that Perfect Writer is sophisticated enough to meet users' word processing needs through the year 2000. This claim may well prove true for a fair number of academic writers, but for others Perfect Writer's complexity and lack of thorough screen orientation make it insufficient right now.

4.4 Expensive Programs

One pays for several things with the more expensive programs. First, they can be presumed to have certain features, like the abilities to vary line spacing within documents, to handle files up to the size permitted by the microcomputer system, to record forced page breaks and conditional hyphens, and to offer several gradations of a special effect (as doublestrike-boldface-shadow print). Because they tend to be complex programs, they often allow users two (or more) ways of accomplishing a function. Second, most of these programs have one or two characteristics developed to an extreme extent: VisiWord's excellent, thorough screen design makes it possible to use it virtually without reference to the manual; WordStar's screen orientation is remarkable for so powerful a program; WordPerfect and Spellbinder offer a host of specialized print functions and "macro" mini-programs. With each, you will be paying for attributes a sufficient number of purchasers have found beneficial enough to keep the program on the market at a premium price. (Remember, though, that the tripartite price division used here is somewhat artificial--e.g., VisiWord is much closer in price to Perfect Writer, in the intermediate category, than it is to WordStar in its own.) Third, these programs can perform more functions in more ways than most cheaper programs. Naturally, they will also be more complex and so more complicated to learn, albeit there is still considerable variation among them in this regard.

Be warned, too, that each of these programs will probably have some "gap" which might prove significant, perhaps crucial, for the kind of work you do. Some cannot do more than one text enhancement at a time ("boldface-underline" is the standard testing instance); others cannot print multiple copies; and still others cannot print foreign language characters or mathematical symbols, or make the creating of them too difficult and impractical to be of any steady use. Still, most academics will probably want to choose a program from this category for their personal use. Word processing is here to stay, so it is probably worthwhile for academics to spend the money and expend the learning time needed to have one of these programs at their disposal. With students it is a different matter: even when very substantial educational discounts put these programs nearly on a cost level with cheaper programs, it will not always be a "bargain" to get a much more powerful program for only a little

more money. If students will not eventually have sophisticated word processing requirements, then they might be better off not having to learn the operation of what will necessarily be a somewhat complicated program.

WORDSTAR

Distributor: MicroPro International
1229 Fourth Street
San Rafael, CA 94901
415-457-8990

Cost: $495

WordStar has been the microcomputer industry's word processing standard for nearly half a decade, a very considerable time to hold this position in such a fast-changing field. While some programs have features noticeably lacking in WordStar, like footnoting and split screen capability (for editing more than one file at a time), no program has proven itself so evidently superior to WordStar that it has been able to dislodge the established leader.

Peter McWilliams alludes to one of the chief reasons for the program's success when he notes, "WordStar is difficult to learn but easy to use" (188). WordStar does not really use the ten function keys--although the version written for the IBM-PC is programmed to provide duplications of common functions via these keys. Most commands (except dot commands) follow the same three-keystroke pattern: the control key, followed by a letter key putting you in the appropriate menu, followed by another letter key effecting the precise function you want to perform. Thus, any line can be centered at any time by putting the cursor on it and typing Ctrl OC. To save a document and return to work on it, you type Ctrl KS; after the save, you press Ctrl QP to return the cursor to the position where you left off typing or editing. The several hundred commands in this three-stroke pattern--or even the several dozen you need to get started--are somewhat difficult to recall and to keep separate at first. Most users I have observed, myself included, also have some initial difficulty making use of the on-screen menus which are meant to show users how to complete the desired function. (The basic problem in this regard would seem to be that it is sometimes hard to guess which menu harbors the function you need--and searching through menu after menu for it can be exasperating.) But after you gain some expertise with WordStar, commands become much easier to remember or locate, because they almost all follow the same three-step pattern. There is no fooling with Alt, Esc, the ten function keys, or the other special keys which so many other programs use, sometimes in very irregular combinations with Shift or each other. In WordStar the Ctrl key does, indeed, "control" matters--it is very nearly the first step in performing every editing and formatting function. Thus, WordStar has the smallest keyboard span of any program: what time is lost to extra keystrokes is more than gained back by the time saved in not having to search out and reach for infrequently used keys.

WordStar is designed to be a screen-oriented program that offers a great many more features than other screen-oriented programs. It may not be as self-evident in its operation as a program like VisiWord, but it is much more powerful and has many more editing and printing options. Print-oriented programs like Perfect Writer and PowerText can do more than WordStar, but require the user to work with a screen display littered with symbols and not at all resembling the text that will be printed from it.

As the user becomes familiar with WordStar, he or she will find that many options have been provided for. WordStar is a program--one of the few--that enacts line spacing on the screen display (a considerable asset, I have observed, for students learning the program, since it helps them envision the relation of screen to printed text). Still, an experienced user might want a first draft printed with quadruple-spacing, which would only put six lines on the screen at any one time, but prefer to work with it double-spaced on the screen display. Arranging this would merely be a matter of setting line spacing to double before entering text, returning the cursor to the top of the file when done entering text, resetting the line spacing to quadruple, and then giving the commend (Ctrl-Q-Q-Ctrl B) to reformat the whole file. After a few moments the file will be ready to be printed out quadruple-spaced. This sequence might seem esoteric to novices, but for even intermediate users it will seem a perfectly logical combination and extension of simpler commands they already know. In dozens of other ways, WordStar anticipates and provides for operations users will find helpful refinements of basic functions.

WordStar is organized around a system of seven menus: Opening, Main, Onscreen, Print, Block, Help, and Quick. The first appears when the user enters the program: it allows for the editing or creating of documents, their printing, deletion, copying or renaming, and a number of other system operations. In other words, it does what the "main menu" does in many other programs. WordStar's Main Menu, which can be left on the top of the editing screen for reference by setting the "help level" to 3 (highest level), provides information on cursor movements, scrolling, deletion, seven miscellaneous functions, and lists the other menus. The last five menus are accessed from the text by pressing Ctrl and O (Onscreen), P (Print), B (Block), J (Help), or Q (Quick). Commands are then enacted by pressing the appropriate single letter. Since almost all commands are fitted into this pattern, WordStar need employ only one other command pattern, dot commands, i.e., those starting with a dot in column one of a command line. These commands are needed for adding certain directions for the printing of documents, like automatic page numbering (starting with any number), a conditional page break (to keep certain units together on a single page), headers and footers, and the direction not to print an on-screen notation. Although dot commands are somewhat complicated--and do present an alternate command pattern--most users are not likely to need the functions they perform when first learning the program, and so they need not be learned until the user has a basic grasp of the program's operation.

WordStar's greatest asset is the clarity of its screen display: what you see on the screen is, more or less, what you will get on the printout. Even veteran users appreciate this attribute, for WordStar offers so many features that it is very helpful to see most of them enacted on the screen, thereby making errors (or even arrangements that do not "look right") easier to spot and adjust immediately instead of having to wait for a printout to discover

them. Moreover, the user will not, as often happens with print-oriented programs, end up putting a whole document in what proves to be, on viewing the printout, an unacceptable format, necessitating dozens (even hundreds) of changes. With WordStar you see all the things you are doing with text as you do them and so can readjust as you go along.

WordStar's commands are sometimes mnemonically organized, sometimes positionally. For example, Ctrl KS is "save" and return, Ctrl KD is save and "done," and Ctrl KQ is "quit" without saving changes. Cursor movement keys are arranged via their placement on the "qwerty" keyboard. Thus, when Ctrl is also held, the diamond a s d f moves the cursor left a space (s) or word (a), right a space (d) or word (f), and up (e) or down (x) a line. Adjacent keys extend the pattern: Ctrl w scrolling up a line and Ctrl z down one; Ctrl r moving up a screen, Ctrl c down one. Part of WordStar's strength is that while it has many complicated (usually three-keystroke) commands, the program does not make learning them any more difficult than they need to be. WordStar is organized and presented consistently and wisely asks user to adjust to its "habit of mind" rather than trying to appeal to a variety of mental patterns.

The user who becomes familiar with WordStar's way of doing things will find, as new functions need to be learned, that they are logical and easy. Margins and tabs can be set in a moment; one file can quickly be read into another; the extra keystrokes for functions like underlining, boldfacing, and overprinting (to add accent marks) soon become such a rote part of the typing that the user is barely conscious of performing them. The user will also find the program has allied features to aid such operations, like a line-by-line margin release command, which is very useful when typing documents (such as résumés) where words are not always to be wrapped; if the user forgets the name of the file to be read into another, Ctrl KF will display a list of all the files on a disk; and it is always possible to check a menu, as a normal part of the operation, to see exactly what letter will start underlining, boldfacing, doublestriking, superscripting, and like operations. Line spacing changes are as well-designed in WordStar as in any program: Ctrl OS followed by 1 to 9 will start designating spacing from the cursor forward; repeating the command with a different number immediately shifts line spacing.

Users will quickly adjust to many WordStar features, like its broken line to indicate page breaks (a feature, incidentally, that can be deactivated), its way of halting wrong operations (Ctrl U, then Esc), its method for printing selected pages of a document, and the number of features that can, like soft hyphenation, be turned on or off. Users will also soon find more efficient ways of doing some things: it is possible to enter the program and put a desired file on the screen with one origination command; there is also a whole series of "quick" commands, often a matter of adding Q to a command (e.g., Ctrl C moves you forward a screen, Ctrl QC moves you forward to the end of a file). Nor will users be restricted to using only one enhancement at a time: a word can be underlined and boldfaced at once, and contain superscripts, subscripts, and overprintings as well. Users will learn certain "tricks" that also improve WordStar's performance; for instance, putting a doublestrike or boldface command at the head of a file will create a dark, clear copy that can make dot matrix-printed documents passable for certain uses. Other advanced features (like spooling) may not prove particularly useful to some operators, but it is certainly better to have them there to try when one reaches the point of understanding how and why they might be used.

It can, in sum, be said that WordStar is one sophisticated and powerful word processing program that any user, save the most antithetical, will eventually be able to learn and use on a high level.

WordStar's long-dominant position on the market--close to a million copies have been sold by now--has resulted in several additional benefits accruing to its users. It is a thoroughly debugged program; many, many programs have been written to be compatible with it (and it has a "non-document mode" for creating materials like data files and computer programs to be used in conjunction with other programs); and it has a number of special capabilities that make possible its interface with a variety of specialized printers, a fair number designed with an eye on WordStar.

Many programs have learned from, copied from, and in some ways improved on WordStar, so it would be tautological to say that WordStar would not be nearly so impressive and popular a program were it introduced on the market today--yet, the fact is that WordStar, for all the considerable advantages discussed above, has a number of significant disadvantages also. Its lack of some features is well-advertised by its competitors: no on-page footnoting capacity, no split screens, no proportional spacing, no delete buffer and, in the main program, no mailmerging capacity. (One presumes that at least some of these features could have been added as the program was updated.) For use in an office or similar setting, WordStar might well be the single most expensive word processing choice, since it divides complementary functions among nine allied programs (all, like WordStar, priced near the top of the market). While some word processing programs can do indexing, sorting, and mailmerging, three separate programs must be bought to accomplish these functions with WordStar. The user cannot even print out multiple copies of a document without the MailMerge program.

Although WordStar is a well-designed program, it has some gaps. The operator cannot shift to continuous underscoring, but must enter an underline in each blank space. It has no "automatic save" feature, so operators must remember to save documents to disk every fifteen minutes or so. WordStar does, though, automatically create backup files. The version of WordStar I tested on the IBM-PC did not activate the backspace-delete key (it backspaced only), but assigned the "backspace-delete" function to the Del key.

Like any program, WordStar has some design features that operators must get used to. For instance, novices often tend to press Ctrl D and Ctrl P, rather than simply D or P, to open or print a file from the Opening menu, and so are puzzled when nothing happens. Similarly, they often forget that after pressing Ctrl OS they must release the Ctrl key before pressing the number key indicating the line spacing they desire; they will be several lines further into the file before they discover the new line spacing has not taken and that this portion of the text will have to be reformatted. Moreover, line spacing other than single-spacing (the default) has to be set each and every time a file is opened, something forgetful operators (including myself) often neglect to do.

Most of its design peculiarities will eventually come to seem natural for users, but WordStar has several design weaknesses that will remain problematical for almost all users. First, WordStar is not screen-oriented when it comes to adding text enhancements like underlining, boldfacing, and

overprinting; instead, it puts four extra, unprinted symbols into the screen text for each enhancement. A word to be underlined appears like this: ^Sthirsty^S. If the justification feature is active, as it will be for most documents, the line on which the word to be underlined appears will, somewhat distractingly, extend four spaces to the right of the justified margin. If the line has several text enhancements--say, several short novel titles, each of which will need to be underlined separately to avoid underlining the commas separating them, the added enhancement symbols and characters will push the end of the line out of sight beyond the eighty-character line limit of the screen display. Moreover, when more than one text enhancement is added to a unit, the screen representation of that unit can become bewildering. For a not-too-farfetched instance, let us suppose you had the French word hotel, as the last word in a sentence that was to be footnoted, and that you wanted to underline it to indicate its foreign language status and to boldface it to emphasize its importance to your argument. It would appear on the screen thusly "^B^Sho^H^tel^S^B.^T1^T". In light of such instances, it is impossible to maintain that WordStar is a completely screen-oriented program.

Another problem with WordStar's text enhancements is that they do not--like Volkswriter's, for instance--shut off automatically at the end of a line or paragraph if you forget to deactivate them. Thus, if you fail to end an underline on page 2, the rest of your text, except what you want underlined (remember it is an off/on "switch"), will be underlined. Since, on average, one such omission is likely every five pages or so, it is often necessary to stand by the printer to watch the printout of a lengthy document and stop it when such an error is noticed. (Having to watch the printout is one reason WordStar's spooling feature is not all that useful.)

Among the sophisticated programs, WordStar is one of the slowest. Holding down a command key, like Ctrl C for going ahead by screenfuls, will soon produce an annoying beeping sound and a string of exclamation points on the screen, indicating the program cannot work this fast. Similarly, when asked to print out a single page from a long file, WordStar can spend several minutes going through the file in search of it. Thus printing, say, pp. 9, 20, 34, and 49 of a file might take twenty or more minutes, since the complete command sequence would have to be gone through for each page and then that page searched out from the front of the file onward.

A problem my students occasionally have--although I have yet to discover the cause--is that they unintentionally direct the program to move many lines of text (often as many as 600 or 700 characters) onto a single line--and then must spend a good deal of time horizontally scrolling out to the end of this "superline," pulling the text back into the document's established parameters one line at a time.

That I have not been able to discover an explanation for this error in the WordStar manual is one indication of the documentation's limitation--but, in general, WordStar's documentation is not as bad as it is usually rated or as incomprehensible as the number of commercially available introductions to WordStar would suggest. The first part of the manual takes the user through a "Short Course," "Intermediate Course," and "Extended Course" (this last dealing mainly with MailMerge and SpellStar). Although the exercises in these "courses" are perfunctory and rather useless, the explanations, if plodding, are generally clear and relatively concise. At the head of each chapter is

its instructional objective. The lesson section is followed by the more extensive "reference manual," really a full "user's manual." In all the documentation, effective use is made of open space, highlighting, boxed charts, sample screens, and other visual aids. The nineteen pages of error messages not only concisely describe the "situation" and the required "action," but also contain the "screen display" of the error message, an attribute that often speeds the linking of the problem to its solution. The index is not as complete as it could be, but does have the virtue of listing commands, alphabetically, in with other entries. Ironically, since it is one capacity WordStar is often scored for lacking, most of the manual is done in double-column print.

WordStar's on-screen documentation is only moderately helpful, especially to the novice. Menus are very full and much abbreviated, and help screens tend to be terse in their explanations. However, the ability to set the on-screen help at any of four levels (from keeping menus on the screen permanently to suppressing them entirely) does allow the operator to make more effective use of what information is available. If WordStar's documentation, both printed and on-screen, has one consistent failing--one cannot really fault it for copiousness, since there is so much to explain--it is that all too often it does not explain why, as well as how, a function should be performed. It is chiefly by setting one's writing in contexts and emphasizing just what features pertain most centrally to such contexts that the better commercial guides to WordStar are superior to, but not substitutes for, the program's own documentation. Their supplemental usefulness may, however, be superceded by the release of WordStar's own on-screen tutorial, a release said to be imminent.

The first sentence in the WordStar manual is "Installing WordStar is easy." Forty-three pages of directions follow. These directions are written clearly enough, but there are far too many of them to pretend that installation is "easy"; at best, installation is "not overwhelmingly complicated." Like much that one reads and hears about WordStar, the manual's opening gambit is a half-truth. At the present moment, it is probably necessary for those looking for a word processing program, particularly for institutional use, to know about WordStar simply because the program is so widely used. Thus, I have gone into its operation at greater length than that of other programs; hopefully, this information will clarify at least some of the half-truths that attach to WordStar.

If cost is not a particular consideration, if a program that will perform sophisticated functions is needed, if that program must also be screen-oriented and relatively easy to learn, if the program is not needed to make text ready for printing things like footnotes, if the program must be compatible with many other programs, and if the program must have proven itself with a wide variety of users over a number of years, then WordStar is still the safest bet. All these "ifs," however, indicate why potential buyers might want to consider some of the other nineteen programs discussed here.

VISIWORD

Distributor: VisiCorp
2895 Zanker Road
San Jose, CA 95134
408-946-9000

Cost: $375

Of all the programs reviewed here, VisiWord is undoubtedly the easiest to learn and operate if one just uses the information that appears on the screen display. (This is not to say that it is the easiest or most convenient program to operate overall.) Dominating VisiWord's design is a rigidly applied classification and division system; to the extent one's mind works along, or can adjust to, such organizational lines, one will feel comfortable using VisiWord. For such users the opening claims in the program's documentation are only slightly exaggerated: "The VisiWord program is so easy to learn and use that you may not have to refer to the User's Guide. The program is virtually self-teaching and help instructions are right at your fingertips." (However, VisiWord's selling slogan--"Writing and Word Processing Made Easy"--conflates, unjustifiably, two functions. No machine can, or should, make writing "easy.") But for users who dislike having to "go" to sub-sub-categories for even relatively common functions--rather, say, than having them as two keystroke commands which can be memorized and quickly enacted thereafter--VisiWord will seem an unduly schematized, and perhaps even ponderous, program.

Although it does not fare particularly well on the number of features it includes for a program in its price range, VisiWord is so easy to start working with and still sophisticated enough to handle most kinds of non-mathematical, English-language documents that it would be one good choice for students who need a program that will enable them to start producing short essays and assignments almost immediately and will eventually allow them to produce relatively complex reports and term papers. Indeed, VisiWord would be a good choice for anyone with fairly complex word processing demands, but who also wants a program that can be learned quickly and operated without having to consult a training or reference manual every few minutes. What one sacrifices in the number of functions and directness of commands in VisiWord, one gets back in immediate comprehensibility and clarity of operation.

VisiWord is organized around a set of menus, sub-menus, and sub-sub-menus. The initial menu, which just offers "create," "revise" or "help," gets one to the editing screen (which is excellently designed), where the main menu is very clearly laid out in a single line beneath the boxed text: "Delete Copy Move Find Layout Windows Storage Print Options ?=Help Exit." These ten choices will eventually lead the user to almost any function in the program. For example, to boldface a letter the "menu path" (the term consistently used in the documentation) is Layout/Char(acter)/Emphasis. The main menu is accessed, as the status line in the thick bottom border of the text box reminds the user, by pressing Esc. Subsequent choices are enacted either by pressing the first letter of the desired menu or command, or by moving the menu cursor, with the arrow keys, to the desired selection and

pressing Enter. (This second method is designed--and works--to give new users a more definitive sense of how the program operates.) A peculiarity of the system is the importance of the F2 (again) key which repeats the function as many times as needed; for example, it must be pressed to boldface each character after the first one. A very useful feature is that every menu includes a "?" option which will call up a help screen explaining all the menu's choices. Normally there are two ways of exiting sub-menus and sub-sub-menus: Esc gets you back to the editing screen, while Quit takes you "back up" to the menu above the one you are in. All this activity can be a bit tedious once the user becomes thoroughly familiar with VisiWord, but the program's rigid, clear organization does make it one of the programs that avoids using Alt and Ctrl keys altogether without having to resort to an equally (or even more) complex set of commands within a variegated character/function key system.

Another immediate benefit of VisiWord, and one which will continue to be a help to experienced users, is the editing screen. As already noted, the text is very clearly boxed, with the status line in the thick bottom border and the main menu choices beneath it. In fact, all four sides of the text box are used for symbols and informational items, which appear there in inverse video. The thick top line contains a ruler indicating tabs, indents, and decimal tabs and has a "shadow cursor," which indicates the cursor's present position on its line. The thinner right and left borders are used for special character symbols, like the theta symbol (Θ) on the left, to indicate where a new format set will take effect, and the paragraph symbol (¶) on the right, to indicate where a paragraph ends. The right border even contains a row of stars on a blank screen, eliminating each as text is put on its line. The status line indicates whether the insert or overstrike mode is active and what line the cursor is on. In addition, on the left the status line always has some kind of "prompt message" (like "Press Esc to enter menu" or "Print: Print the document on a printer") meant to clarify the most likely next step at any juncture. Moving around a VisiWord document is fairly easy, with Home moving the cursor to the beginning of a line the first time it is hit, to the top of the screen the second time, and to the head of the file the third time; with the End key performing these maneuvers in the opposite direction. The one small drawback of VisiWord's otherwise excellent editing screen is that it allows only twenty-two, rather than the usual twenty-four, lines on the screen at one time.

VisiWord does employ the function keys, in what might first seem a rather miscellaneous manner, but which will prove, for most users, an effective complement to the program's tight menu structure. The ten function keys--actually nine, since F7 is a "dead key"--are meant to be quick ways of implementing what are likely to be among the most commonly enacted commands. Thus, F10 centers a line, F1 gives a decimal tab, F4 indents a line, F3 starts a new page, and F8 underlines a character. (This last function can also be performed through the menus, a fact which makes one wonder why VisiWord's designers did not utilize the "dead" F7 key for duplicating another fairly common function like boldfacing.) There is a template, indentifying VisiWord usages, to fit over the function keys, but the user will probably find this is not needed after a few hours steady use of the program.

Several other VisiWord design features will be helpful to most users. Its page break symbol, a line of diamonds, is very distinctive; text moves up

only one line at a time, rather than half a screen, when typing reaches the bottom of the screen (an attribute that helps new users conceptualize the relation of screen text and printed text more accurately); and the program loads a "format sheet" (which can be created or pulled from "storage") with each document, a decided benefit for steady users who work in a variety of specialized formats.

Despite the clarity of its overall design, VisiWord is curiously incomplete in the design of several of its important features. It uses delete-backspace and Del for deleting characters, and has F5 for deleting a line and even F6 for deleting a line from the cursor forward, but it has no command for deleting a single word (which is surely a more common function than deleting all or part of a line). Insert is, as it should be, the default, with the Ins key an easy way to toggle back and forth to strikeover--but inserting each character causes the text to start reformatting itself accordingly, meaning that the text following the insertion is jumping, very distractingly, this way and that as characters are inserted.

VisiWord permits line spacing of ½, 1, 1½, and 2--but nothing larger, so one cannot have the triple- or quadruple-spacing that is often helpful for the hand editing and revising of early drafts. F9 is the "undelete" key, but it only works with certain kinds of deletions, namely, those made via F5 or F6--and so is of rather limited use. The program does provide on-screen justification, but, as the manual notes, "the spacing of justified text may look slightly different on the screen from justified printed text." The overstrike feature can be used to put in accent marks for foreign language characters, but that character cannot then be underlined, as it will often need to be.

Other design features have been worked out thoroughly and logically: including and appending documents is easily accomplished; material can be moved, copied, or deleted either by "text" or "column" (in general, columns are easy to work with in VisiWord); since the user must designate a name at the point of saving a document, preserving each draft of a document is simply a matter of giving each a different name; and VisiWord has a full window capability for viewing documents simultaneously and for shifting between the editing of them. Its file maintenance sub-sub-menu provides a flexible list of operations for keeping files in order; its warnings when space is "low" help the user move to a new disk when necessary; and its spooling capability speeds operations when time is at a premium. VisiWord has the added, but expensive ($195), benefit of being able to work with VisiSpell, one of the most useful spelling checkers around, as well as with VisiCalc and VisiFile.

Like some of its features, VisiWord's documentation is good but incomplete. The manual comes in an attractive, manageable black looseleaf binder, is effectively organized (pages are clear and make good use of open spaces), and very well-designed graphically (although it could use a few more sample screens). What the manual most conspicuously lacks is full explanations of each and every function (i.e., more than just a single sentence indicating what a command will do). Sometimes a command is self-explanatory, but often a short paragraph is needed to clarify exactly what VisiWord means by a particular command, how the command will be enacted on the screen, what alternate commands the user might find more helpful for a specific effect, and other bits of information.

The manual's index is more like a book index than those in most program manuals; most entries appear as subheadings under a main heading (e.g., "deleting" has thirteen entries under it), rather than as part of a long alphabetical list that scatters related entries. Thus, it is easier to find information when you only have a general notion of what category in which to look for it. Unfortunately, though, the index does not italicize or boldface the most significant page references, a conspicuous failing because the first or even the second mention of an item is typically not the most complete one, forcing the user to waste a good deal of time checking each reference until the "complete" one is found.

Several features of the manual do, however, make very good sense. Throughout it has "Warnings" and "Notes" clearly set off from the main text--these efficiently handle those additional bits of information and explanatory cautions which otherwise could clutter the main text and confuse the reader trying to discern the most usual form of a procedure. The appendix of error messages is relatively comprehensive and clearly written, and it contains suggested solutions for the problems. The "Quick Start" course, a twenty-four page pamphlet in the front of the binder, is one of the best available. Neither too simple nor too complex, it takes the novice through the effective creation of a sample document, speaking in a rational, informative tone (not, thankfully, a self-complacent or overly enthusiastic one), making very effective use of illustrations and sample screens, and providing, at its end, a very helpful, clear chart of all the basic operations covered in its exercises. After doing the "Quick Start" course, the learner will have a good basic grasp of the program's operations and be ready to approach its less common and more complicated functions comfortably and intelligently. In doing so, he or she will probably find the information provided by the on-screen menus and their respective help screens quite thorough, but the supposedly more extensive information to be found in the manual will sometimes be just repetition of what has been said on screen.

Those who have subtle and varied print requirements will not want to use VisiWord, for by avoiding embedded commands the program severely limits the range and variety of its print options. It will not do on-page footnotes, for instance, and its footer capability cannot be used for makeshift footnotes if any is over three lines. More significantly, it allows underlining or boldfacing or superscripting or subscripting or overstriking of any character, but no two of these together, a weakness that, as already noted, limits its use in many foreign language writings--and undoubtedly in some mathematical and scientific writings as well. Unless the user takes special precautions, losing a file is always a possibility.

Still, VisiWord should be given serious consideration by academics working almost exclusively in English. It is a program that will enable the user to start writing "real" documents on the word processor after just a few hours--and it is powerful and sophisticated enough to handle most kinds of writing. Similarly, it will meet most of the word processing needs of students who will be working chiefly in English. For large classes, where the instructor will not have a great deal of time to devote to the questions and problems of each individual student, VisiWord would probably be, barring cost, the best choice, since it is the most self-teaching and most "self-evident" of the more sophisticated programs. It is also, more rather than less, screen-oriented: e.g., that characters to be underlined appear in inverse

video is only a very slight "discrepancy" from printed format. For those who can afford it, VisiSpell is one allied program well worth considering. For anyone who does a fair amount of writing and who is very uncertain about whether he or she can quickly learn and effectively use word processing, a few hours work with VisiWord should provide the answer. VisiWord's overall design might not appeal to all users, particularly experienced word processors who do not need such a rigid schematization to understand and operate a program, but many users will find the program both easy to learn and sufficient to meet their long-term word processing needs.

SPELLBINDER

Distributor: Lexisoft, Inc.
Box 1378
Davis, CA 95617
916-758-3630

Cost: $495

Spellbinder, which, despite its somewhat misleading name, is a full-featured word processing program, rewards perseverance. Although it is one of the older programs around--its original version dates back to 1978--its program manual remains a most forbidding and unattractive, yet compulsory, entry to what is, to be sure, a complete and difficult program, but also a very powerful and generally sensible one. Because Spellbinder is not a screen-oriented or "screen-evident" program (like VisiWord, for instance), the user must first come to some terms with the manual.

Since Spellbinder has been adapted, not designed for the IBM-PC, IBM users will encounter an eight-page supplement, packed with information, to get through before they even reach the manual proper. Once there, they will find a "manual--including all text and headings--[that] was printed using the Spellbinder™ word processor system and a Santec S700 printer." They will also find a perfect, if unwitting, demonstration of word processing's limited ability to approximate the layout, look, and visual variety of standard publishing formats.

Crammed with information in small, rather unattractive print, the Spellbinder manual resembles those automobile manuals which describe every section of the car down to its nuts and bolts but which neglect to tell the user how to change a tire. The manual's eleven substantial chapters--which are divided into as many as thirty-five, likewise substantial, sections--seem endless and bewildering in their explanations. On almost any page the user is told things like this:

> If you want a command chain to run entirely without intervention, you must be sure it does not contain an i command which normally requires a response. For example, if there is a da in the command chain, Spellbinder will

> stop at that point and ask 'REALLY?' The operator must respond to that question before the command chain will go on.

The experienced word processor who comes to the Spellbinder manual may be able to make better sense of each chapter by working backwards, i.e., going from the general review section at its end, to the function "summary" at the end of each section, and only then into the full-dress explanations themselves. The novice will just have to plug ahead page by page, although two features may help with the learning. The "Quick-Reference Guide" (Appendix A) provides a fair overview of the program by listing its many Edit Mode and Command Mode functions (see below), as well as frequently used "table settings," dot commands, and in-line commands. The exercise the learner is given early in the manual--a specific letter to type, format, revise, and print--is a bit plodding, but it does demonstrate that it is possible to use the program understandably. The help screens in Spellbinder are dense and much abbreviated, but learning to read and use them is at least easier than learning to read and use the manual. In addition, Spellbinder can be configured to keep certain kinds of operational information on the screen, although the novice will probably need an experienced word processor to set up this feature.

The user who perseveres through the slow and often confusing process of learning Spellbinder will then be operating a program which, like PeachText, is particularly (and usefully) rich in print options--and which also has some features that make editing noticeably easier. Spellbinder does true proportional spacing, has a whole series of "dynamic formatting commands" (like vertical tab, pause, backspace, firm hyphen, and negative line feed) to aid in the creation of idiosyncratic and irregular formats, has a separate "table" just to set page numbers and titles (in a variety of ways), allows four kinds of justification, and permits the printing of files directly from the screen rather than necessitating they first be saved to disk. Spellbinder's on-page footnoting capacity is very sophisticated, allowing for automatic formatting and numbering, and even the splitting of footnotes between pages. Alternately, footnotes can be printed at the end of a document, a benefit for those who do not need on-page footnotes but who prefer to write footnotes directly after they write the text to which footnotes refer. (To do this in most programs, the user would have to create a separate file, go to and from it for each note, and append it to the text file when done writing.)

Another noteworthy asset of Spellbinder is its speed. On average, it probably performs functions faster than any other program analyzed here--just press End and see how quickly the cursor will move to the end of even a long file. Smaller cursor movements, of which there is a useful variety, are equally rapid. Although speed of operation may be more important in a secretarial situation than in an academic one, all users will eventually encounter situations where they need to jump around and about in text to perform a series of small operations and so will appreciate a program that allows them to do so rapidly.

Spellbinder provides a number of advanced editing features that can eventually--one would not want to bother with them at the outset--make revising and changing documents all the easier. For instance, one can set the

"cursor mode" to character, word, sentence, paragraph, or mark and move the cursor or delete text according to the designated unit. A more immediately useful command is "scan" (Ctrl s), which moves the cursor to the end of the line when pressed once and to the beginning when pressed a second time. Spellbinder has a "holding buffer" where deleted text (or text to be moved) is put--and from which it can be easily recovered if a mistake has been made, a backup feature that encourages new users to try operations they might otherwise avoid and so not learn. The program will do decimal tabs, write a designated part of a document to disk (a feature not found in many programs), and put unprinted notations in a text; it permits horizontal scrolling (lines can be up to 158 spaces long) and offers a very powerful search and replace capability (allowing, among other operations, "wildcard searches"). Although some users will not like the overall design of its insertion mechanism, most will find that the space opened for large insertions makes entering them a clean procedure.

The overall design of Spellbinder is thoughtful in many particulars, if complex when the sum of its features is totalled. It will readily display the amount of space left in memory; it can be precisely configured to the microcomputer system being used; and it offers, as part of the main program, "macro" mini-programs that make Spellbinder, for certain kinds of work--it is indicatively subtitled a "Word Processing and Office Management System"--the single most useful program available.

The greatest flaw in Spellbinder's design is the exclusivity of some functions. For instance, one cannot enact more than one enhancement at a time (underlining, doublestriking, superscripting, etc.), a surprising limitation for so generally powerful and supple a program. More importantly, Spellbinder is a dual-mode program: the user must constantly, and disconcertingly, toggle back and forth between the edit mode and the command mode, a feature which strongly (and unfortunately) discourages spontaneous revision and rearrangement in favor of a more regimented division between writing and revising. Another inhibiting feature is that you are not able to make insertions within insertions, but must cycle back through a whole series of procedures; for example, if you forgot to type in "three" before the insertions "cats," you cannot simply move the cursor back in front of "cats" and type in "three," but must complete the insertion and do "three" as another insertion. Similarly, although Spellbinder provides a "preview mode" that allows the user to see a document as it will look when printed--a considerable help for a print-oriented program like Spellbinder--it does not allow the user to stop the text and enter changes, a limitation that severely undercuts the value of this feature.

Spellbinder's lack of screen orientation is rather extreme even for a print-oriented program. Text borders are not outlined, and the program only divides a file into pages when it is ready to be printed. As for removing text, Spellbinder did not make the delete backspace key work on my machine, although the manual indicates it should have (and presumably will on your system). It is, on the other hand, a bit too easy to delete largish sections of text accidentally with Spellbinder.

Some operations, like saving a file, involve quite a number of steps, although they are not hard to master. Several large-scale commands are similar enough to lead to their accidental and perhaps disastrous confusion--

e.g., "Com. q" lists files and "Com. gd" writes files to disks, but "Com. qd" deletes files. In all, Spellbinder's command structure tends to be more thorough than efficient, a characteristic epitomized by the fact that the command sequence to print a second copy of a document requires all but one of the steps needed to print the first copy.

Like WordStar, Spellbinder antedates the IBM-PC by several years and so makes only supplemental use of the IBM keyboard (i.e, employing function and other keys as alternate ways of performing functions). Unlike WordStar, however, it makes significant use of these keys: F7-10 each gives you another whole set of operations using the function keys again. For instance, F8-F9 "deletes from cursor" (= Com. d [Ret]), and F10-F10 "displays file status" (= Com. st [ret]). Spellbinder also involves several other special IBM keys, like "7" on the numeric keyboard as a duplication of the "scan" command. Some of the alternate keystrokes will be chosen by most users; for instance, it is clearer to use the four arrow keys on the numeric keyboard to move the cursor than it is to use the Com. set of k (up), j (down), 1 (left), and h (right), which puts the cursor movement commands in a line rather than in a more referential diamond pattern. And while any IBM user must still enact a number of Spellbinder commands via the Command mode, that user, if he or she prefers, can employ the function and other special keys for almost all procedures performed on a regular basis.

Spellbinder also uses dot commands--in standard ways (.e forces a new page) and in less usual, but not uncommon, ways (.c centers a line). Like Volkswriter, Spellbinder requires the insertion of a whole new "format" to change line spacing, a cumbersome procedure, but one that allows margins, justification, and other features to be altered along with line spacing.

Spellbinder's strongest selling point--what may make it worthwhile to suffer through its manual to learn what, at best, remains a fairly difficult program--is its "macro" feature. These are mini-programs which require only a few key-strokes to perform complicated, individualized operations with files. Lexisoft provides fourteen macros with Spellbinder, and they can be saved in configured versions, an option which enhances their specific value. Lexisoft also offers a macro program manual (the cost is not stated), so those with some programming expertise can create their own macros. The fourteen prewritten macros, each of which is a separate text file, range from those that perform single, simple functions (PAGENUM prints page numbers greater than 255) to those that perform certain complex functions typically found in more sophisticated programs (BOILER does boilerplating) to those that perform functions that are often done via peripheral programs (MLABEL prints mailing labels from an address list) to those that perform functions that other word processing programs, by themselves, cannot do: CUESORT will extract items from a list by predefined categories, like zip codes; ALPHA will arrange a list in alphabetical or numeric order; ADDIT will add columns and rows, automatically entering their sum; CALC will perform addition, subtraction, multiplication, and division of numbers in a text or of those entered from a keyboard. Other macros handle operations like double-column printing, batch printing (one file after another automatically), mailmerging, creating and entering data onto forms, inserting "keyphrases" with a single keystroke, and alternating tabular text. Most of these functions, by their very nature, will pertain more to office than personal work on a microcomputer--but the option of learning "M-speak" to create macros for one's own use is certainly an

attraction of Spellbinder for anyone who will be relying heavily, and in varied ways, on word processing. Given its macro programming option, Spellbinder is perhaps the most potentially powerful microcomputer word processor on the market.

Another interesting aspect of Spellbinder is that for an additional cost it can be had with the allied program, Electronic Webster, which sounds rather like a housecleaning robot, but which is actually a fair-sized spelling checker (50,000 words) and basic grammar checker. Its spelling dictionary is large enough to be of some real use, and operators can scroll backward and forward through its listings, making it resemble a printed dictionary in one productive way. Its grammar checker will search out simple grammatical and composition errors: it can spot double negatives, identify wordy phrases, and indicate misused words. The grammar check can also be configured to spot usages with which the user typically has trouble. When a word or phrase is identified by the grammar checker, the operator has eight choices, including the option to correct it (would that Spellbinder's "preview mode" allowed this), mark it, omit it, or skip it thereafter. Since Spellbinder will most frequently be used in office settings, Electronic Webster should help to eliminate the verbose and inaccurate prose often emanating from business and institutional offices, particularly when a document has been constructed and written "by committee."

As should be more than evident by now, Spellbinder is hardly a word processing program one would want to use with students. Neither is it one most academics would pick for their own work, unless that work was so complex and varied that they would need to get deeply involved in specialized word processing to accomplish it. But Spellbinder is very definitely a program one would want to have available in most large (and some small) offices, along with several users who had taken the considerable time needed to learn it thoroughly. Employing Spellbinder for simple and straightforward word processing tasks is definitely a matter of "overkill," but choosing it to perform complicated, specifically tailored word processing (and related) tasks might not be just a good choice--but the only adequate choice.

EASYWRITER II

Distributor: Information Unlimited Software
2401 Marinship Way
Sausalito, CA 94965
415-331-6700

Cost: $395

Like Volkswriter, EasyWriter II is clearly a second-generation microcomputer word processing program--and, in general, a good one. In assessing the first generation of programs, EasyWriter II's designers--actually Basic Software Group, Inc. of Vancouver, which sold it to IUS--obviously decided that screen orientation would be their top priority;

thus, except for the fact that text always appears single-spaced on the display, this is the most completely screen-oriented program available. Although its command structure is not the best ordered or easiest to learn, these drawbacks are much compensated by the fact that virtually every command will be enacted on the screen, thereby allowing the user to spot and correct erroneous commands instantly--or even redo commands which do not result in the expected "look."

EasyWriter II's screen display goes so far as to include headers, footers, and page numbers. In return for so complete a screen reflection of the text to be printed, the user must forego certain features, like on-page footnotes, that the typically more powerful but decidely more complicated print-oriented programs offer. (Although there are some very powerful screen-oriented programs around, such as WordPerfect, that will do on-page footnotes, none of them can quite equal the sum of features offered by the most powerful print-oriented programs.) EasyWriter II is also curiously lacking a few features, like soft hyphens, which have nothing to do with the limitations imposed by its screen orientation. On the other hand, it has a few uniquely simple and effective "new" features--e.g., the "Page Cut" function easily readjusts a page break up to nine lines in either direction.

The program's thorough screen orientation, together with its several built-in safety features for the editing of text, makes it one strong candidate among the sophisticated programs for general classroom and student use. For faculty just learning word processing, but still in need of a fairly powerful program, it would also be one good choice--but faculty with some word processing experience might want a program that offers more (or different) features or, conversely, is easier to learn and operate.

One of the potential difficulties with EasyWriter II is that its overall design, while perfectly coherent, displays no strong, obvious organizational pattern or "generating logic," and so learning one set of commands is not necessarily an aid or clue to learning another set. (One of the drawbacks of modeling operations on different kinds of programs is that one can end up with rather a discontinuous set of operations in one's own.) Although the documentation claims that the function keys control editing, this claim does not really hold for their primary functions (i.e., what they do as single keystrokes). F2 is Help, and F3 through F7 set the "mode" at character, word, sentence, line, or page for operations like cursor movements, insertions, and deletions. F9 and F10 work in conjunction with F3 to F7, moving the cursor to the previous or next unit that has been designated as the "mode." (These keys are complemented by Shift F5 and Shift F6, which change the mode to paragraph and block.) Together with F1, which "Acknowledges" an error message (not, so far as I can tell, a too common occurrence), and F8, which "Adjusts" (i.e., reformats) text from the cursor forward, F1-10 is a rather miscellaneous set of nonediting, preparatory editing, and actual editing commands.

Although the manual describes EasyWriter II as a "menu-driven" program, this designation is accurate in only a limited sense. More common operations start from the "Main Menu" and less common ones from the "System Function Menu," but unlike programs such as WordStar and VisiWord, EasyWriter II does not extend this organizational pattern through a series of sub-menus. The program may be "menu-driven," but it is not "menu-operated." Besides the usual choices, like "Edit" for opening or retrieving a document, the main menu

contains several that relate to EasyWriter II's particular way of handling files. Since files are grouped into "file folders" so related ones can be kept together, there are choices for opening a file folder and for listing the documents in one. There is also a choice for noting the date on the document to be written or edited. (EasyWriter II is a very orderly program.) The "Paginate Document" choice allows for the repagination of documents, and the "System Functions" choice moves the user to the System Functions Menu (on similar systems necessitating removal of the main disk and insertion of the "Housekeeping" disk). This menu permits functions like copying documen s, editing the default ruler and system parameters, and reconfiguring the program to a printer--its placement ancillary to the main menu makes good sense and allows the user the option of making "global" changes before moving to the document where they are to take effect.

EasyWriter II is designed to be not only screen-oriented, but also to be what might be termed "screen-helpful"; namely, it tries to provide a good deal of information on the screen. Its status line, which is actually three lines long, is typical in this regard. The first line shows the present "mode" (Char, Word, etc.), displays prompt messages, and indicates the percentage the diskette is full. The second line indicates the cursor position by space, line, and page; the document name and number; any shift to special fonts (like boldfacing); and whether the current page is full. The third is a ruler line showing present margins, tabs, and decimal tabs and indicating the center space within the current margins. When you start a file, a line appears beneath the line on which you are entering text, moves down as text moves down, and disappears when the screen is filled--a device evidently meant to make the user aware that the screen is, like a container, being "filled" by text.

Most users will find that EasyWriter II does live up to its name concerning a number of editing tasks. Using Home with the four arrow keys will immediately move the cursor to the top, bottom, right, or left of the screen. Centering text is quickly done (Alt F2) and immediately visible. An "R" appears in the rightmost space of the screen on any line where a ruler has been inserted--and up to eight rulers can be created and named so their insertion is just a matter of a few keystrokes. Special fonts (i.e., enhanced characters) are neatly grouped as Alt key functions (the program reserves the Ctrl key for use with EasySpeller) and mnemonically designated functions, e.g., B giving boldfacing, U(p) superscripting, and "_" standard underlining mark. (Of course, these enhancements immediately appear as such on the screen text.) Most users will find that the requirement to reformat text, via F8, after making changes, is a worthwhile extra step that avoids the visual confusion of text having to reshape itself as insertions and deletions are in the process of being made.

A few of EasyWriter II's design features might seem peculiar. Line spacing is done indirectly, by designating on a ruler the number of lines to be printed per inch; thus, "6" gives you single-spacing, "3" double-spacing, etc. As the cursor is moved over text, or a character is repeated, the machine emits a low clicking sound, a feature which can be deactivated if the user finds it more of a distraction than an aid for drawing attention to such operations. The program offers the user the chance to enter four optional pieces of information concerning each document--the user must enter file name and EasyWriter II automatically gives it a number and indicates its page

length--information which might be useful in an office setting: Author, Operator, Date Created, and Date Edited.

Besides the print command for a whole file, the program offers Shift F7, which prints the page the cursor is on (the whole page, not just the portion of it on the screen); since EasyWriter II has a command that quickly moves the cursor to any designated page, this optional command means single pages of a document can be printed without the more complicated and time-consuming procedures (closing a file and then going through a series of print commands) that other programs typically require to perform this operation. EasyWriter II offers other thoughful features as well; its rather esoteric talk of "hot zones" and "balance points" means simply that with a document where the right margin is unjustified, the program nonetheless adjusts text so that the variation in line length is not too great and so that the right margin is only so "ragged" and no more.

As is probably evident by now, EasyWriter II's greatest strengths, starting with its full screen orientation, lie in the editing area. Its ability to move to any page in a document makes intermediate and long-range cursor movements very easy and quick. (Ironically, EasyWriter II shows up on some checklist reviews as lacking a command to move the cursor to the top of a file simply because it does not have a discrete command for this operation, relying instead on the user designating "1" as the page to which the cursor should be moved.) Its ability to perform insertions, deletions, and other operations by "modes" ranging from a character to a paragraph or block will be very efficient for users who can adjust to thinking and working in such terms. (Others may continue to find it awkward, or may be put off by things like having to decide before an insertion is made just how long it will be.) EasyWriter II offers a "Tag and Merge" feature, which permits various parts of a document to be marked for copying to other places within or without a file, thereby making abstracts and condensed versions of documents very simple to construct. Headers and footers, which can be more than one line, are created by calling up and filling in a very clear "form," and page numbers can be done in a variety of formats. The program also offers some fairly uncommon options, ranging from double underlining to the ability to paginate a document so that an equal amount of text appears on each page. It even offers an "ignore case" command for obviating certain operations in designated circumstances. And while the insert mode is not the default, the space that is automatically cleared when one toggles into Ins and the reformatting that automatically occurs when one toggles out certainly provide compensatory benefits for the extra "work" insertion entails.

In the short run, EasyWriter II is particularly strong on backup features. Shift F2 will cancel the changes on a page and return it to its original state. The System Functions Menu contains an option that checks the "integrity" of a disk, so the user can be sure before recording a text on it that the disk is functional. (Since floppy disks are fragile and do wear out, it is very good to know ahead of time that one is not about to enter a text on a "bad sector" and so make it unrecordable.) EasyWriter II even allows for the printing out of documents with all their rulers inserted, so the user can quickly check to see where an error has been made when a document does not turn out looking as expected. In the long run, though, the user must be careful to back up documents on a second disk, since EasyWriter II does not

create backup files. (Nor does it have a particularly easy method for preserving alternate drafts of documents.)

EasyWriter II, as befits a relatively new program on the market, interfaces explicitly with a number of the more popular printers and will support options like proportional spacing. It can do spooling and has a command to pause printing for typeface changes. For $70 one gets a year of EasyWriter II's telephone support service, which many users think the best (and most gracious) available.

EasyWriter II, even more than other programs, seems to have been designed with a very particular kind of user in mind: one who needs powerful and supple editing features, but not highly specialized print capabilities; one who needs a screen display that is easy to read and almost exactly reflective of the text to be printed, but who is willing and able to learn a relatively complicated and variable program. As a result, EasyWriter II lacks some of the advanced features one often finds in print-oriented programs: capabilities for double-column printing, on-page footnotes, right-justified left-ragged margins, and unformatted printing. Neither will it create indices or tables of contents. As for editing, it cannot do split screens and has no command to make an operation repeat "n" times.

Beside these lacks, EasyWriter II has some design weaknesses. It can do "half spacing" but not true half-spacing: for instance, one sets the line spacing to four lines per inch to get "1-½ spacing," but what one actually gets is something closer to double-spacing ("3") than single ("6"), rather than something midway between the two. Moreover, using double-spacing (or larger spacing) requires a series of operations to insure that the program will still paginate a text correctly. And, as already noted, the fact that all text appears single-spaced on the screen is a significant compromise of the program's otherwise excellent screen orientation. (The user should at least have the option of adjusting the screen display to reflect actual spacing.) While "Char" is the "mode" default when a file is called up on screen, the program does not automatically return to "Char" after each operation--cursor movement, insertion, deletion, etc.--by "mode." Therefore, the forgetful user could subsequently delete a whole line or even a whole paragraph by accident when he or she only meant to delete a single character.

Several other aspects of the program might prove a minor annoyance, like the fact that only twelve file documents can be listed at once. A potentially more important drawback for some users is EasyWriter II's incompatibility with other programs and equipment--it was designed for self-contained use on the IBM personal computer line.

The EasyWriter II documentation is both good and bad--i.e., it makes for clear, understandable reading as one goes through it to learn the program, but it functions poorly as a reference guide to the program when one goes back to it. Its index is, in virtually all respects, insufficient. (Just, for instance, try to discover what the "Acknowledge" key, F1, does.) Part III is, along the lines of the original EasyWriter manual, an alphabetical reference guide to program features and operations, but unless the information you are looking for obviously falls under one of this guide's entries you will have a very hard time searching it out. Also, the binder provided with the whole

EasyWriter II "system" (which includes EasyMailer and EasySpeller) is too small--pages are very difficult to turn and often tear out of the ring binder.

EasyWriter II's documentation is generally better instructionally than informationally. While its help screens are rather brief and may, unfortunately, require the user to turn back to the manual, its thirty-one "page" on-screen tutorial is excellent, providing clear explanations of the program and offering exercises of just the right complexity to give learners a realistic sense of how the program works. Likewise, the manual contains effective "hands-on" exercises, including a set that can be used along with the on-screen tutorial. The tone and approach of the tutorial is typified by an injunction that appears early on: "Experiment for awhile on your own with boldfacing and underlining." Compared with the many tutorials which are regimented and highly directive, this one is refreshingly casual and does allow the learner the freedom to practice operations until he or she has a basic grasp of form.

All in all, EasyWriter II is a program for users who agree with its designers that screen orientation and editing flexibility should be the paramount characteristics of a word processor, and that one can forego many print features and tolerate a fair amount of operational complexity to have these characteristics in full measure. Those who want or need more print power or prefer ease of operation to ease of reading the screen display will want any one of a number of other programs. But those who want the particular combination of advantages EasyWriter II offers will find this program has no near competitors.

WORDPERFECT

Distributor: Satellite Software International
288 West Center Street
Orem, Utah 84057
801-224-8554

Cost: $495

A good deal about WordPerfect may be surmised from the keyboard template that comes with the program: an L-shaped plastic piece which fits over the function keys and extends along the top of the keyboard from the Esc key to the +/= key. In all, it designates twenty-three keys to be used, by themselves and with Shift, Alt, and (occasionally) Ctrl, to enact thirty-nine commands. Most of the designations are evident, like "Cancel" and "Exit/Restart"; a few obviously lead to a series of further choices, like "Print Modes"; and several clearly refer to operations in WordPerfect's particular terminology, like "Set" and "Def Inv Macro." What the template tells the user is that this is a complex program which freely makes use of a variety of command key structures. What it does not indicate is how well-designed the program is overall: while by no means easy to learn, WordPerfect is logical and consistent, which makes it easier to learn than it

first appears. Triple-tier commands like Ctrl N-Ctrl P-Ctrl P may never become simple, but they will make sense to the user who gets to know the program.

Of all the screen-oriented programs, WordPerfect probably comes closest to matching the capabilities of the best print-oriented programs, thereby making it an attractive choice for faculty and advanced student work. (Despite its commendable screen orientation, WordPerfect is too complex a program for students to learn if they will only be using it for fairly simple and straightforward kinds of documents.) Like WordStar, but in different respects, WordPerfect compromises its screen orientation in one large and a few small ways. It will put the text on the screen spaced as it will be printed, and it will enact centering, underlining, and boldfacing on the screen. However, justification does not appear on the screen. Moreover, its line advance key, which moves text up or down half a line (a particularly handy feature for mathematical equations), does not adjust the text on screen. And when a character is overstruck, to put in an accent mark, the character itself disappears from the screen. As for editing, WordPerfect has one small compromise: it does not allow split screens, but does offer "dual document" editing, which puts two documents in the active memory and allows the user to alternate (and move text) easily between the two.

Because WordPerfect's documentation has certain peculiarities, it very much restricts the way in which the user approaches and learns the program. (WordPerfect is very definitely a program one should not try to learn by working with the screen display, so the documentation is the only real entry to it.) The manual refers to keys only by the program's own name for them, and so the reader must constantly consult the foldout reference card, which contains 129 commands and a keyboard diagram, or the template, to discover that "Align" is F3, "New Page" is Ctrl-Return and that Esc is the "repetition counter." Moreover, the manual has the disconcerting habit of always referring to a command "key" even when two or more keys are involved. For example, the "Cancel" key is not, as one might expect Esc, but rather two keys: Alt and F1. With so many commands in such a variety of patterns, even experienced users of WordPerfect will find themselves shuffling back and forth between the manual, the command list, and the keyboard diagram, trying to ascertain just what key or keys WordPerfect is referring to when it speaks of the "Math Off/On Key," the "Display Print Key," and dozens of others.

Another peculiarity of WordPerfect's documentation to which the learner must adjust is the fact that the manual is organized basically as a section-by-section reference guide to its operations. After an opening chapter on "Basics of WordPerfect," in which elements are discussed in the sequence a user would normally encounter them in creating and editing a simple document, the remaining chapters are alphabetically arranged entries, each given its own page(s), relating to the chapter's subject. The second chapter, for instance, is on "Typing" and contains twenty-four entries, running from "Alignment Tabs" to "Widow-Orphan Protection." Fortunately for the learner, these entries are usually concise and clear, relegating additional information and qualifying circumstances to a "Notes" section at their end; almost all the entries also contain a basic exercise so the learner can practice the operation described. Those with some word processing experience will soon be able to jump about the manual, finding information as they need it. (The excellent index, with multiple entries for most items, is a great help.)

Those new to word processing will take a bit longer to get used to "rearranging" entries in the order in which they will need to read them and utilize the information they contain.

WordPerfect's on-screen documentation follows much the same function-by-function pattern as the manual: pressing the help key (Shift F3) before pressing the key(s) for a function will put an explanation of that function on the screen. In all, WordPerfect's documentation seems designed more for long-term than short-term use: the new user will lose time at the beginning in patching together an overview of the system, but that time will be gained back later when the user is quickly able to find precise information on and directions for the many relatively uncommon, but occasionally very useful, functions the program can perform. Like the program itself, WordPerfect's documentation becomes much less forbidding and much more usable as times goes on.

WordPerfect touts its "clean screen," meaning that apart from the status line (containing information on the document's number, the cursor's position by page, line, and column, and a space for messages) "all the other lines are used for your document." Here, again, new users may find the program's design not particularly helpful; the unboxed look of the text (only a space is used to separate it from the status line) takes a bit of getting used to. With time, however, they will learn to appreciate a screen display that is not cluttered with information they have already thoroughly assimilated. A feature that may take more getting used to is WordPerfect's unique way with line ends and hyphenation. Since justification does not appear on the screen, the program has a "hyphenation zone"--preset to five spaces either side of the margin, but it can be adjusted--and prompts the operator to hyphenate any word which falls outside the zone, so as to prevent too ragged a look with unjustified text or too spaced-out a look to some lines with justified text. Since the program does allow for soft hyphens, this feature is useful when trying to create a very neat orderly document. And if the user finds it distracting, as it might be when typing in a first draft of a document that will be altered radically, it can effectively be shut off by making the zone so large that a word will almost never fall outside it.

WordPerfect's cursor movements illustrate the best features of the program's design. Like those in several other programs, they are organized around a simple pattern of "additions": e.g., adding Home once before pressing a cursor arrow key will move the cursor left to the beginning of a line, right to its end, up to the top of the screen, or down to the bottom; pressing Home twice first, then the up or down arrow will move the cursor to the beginning or end of the file. Similarly, adding Shift before striking the left or right arrow key will move the cursor a word in the indicated direction (although you must look at the keyboard diagram to figure out that the manual is telling you this). WordPerfect uses the + and - keys on the numeric keypad for moving text a screen at a time, reserving PgUp and PgDn to move a full page at a time. (Most programs require the user to move through pages a screenful--i.e., a little more than a third of a page--at a time.) Adding Esc before the arrow keys or PgUp/PgDn will move the cursor eight spaces, lines, or pages in the indicated direction. Adding a number between Esc and one of these keys will move the cursor that number of spaces, lines, or pages. The "Go To" command (Ctrl-Home) will move the cursor to any designated page, or to the top or bottom of the present page, or even to the next occurrence of any

designated character. The excellence of this whole set lies not just in the variety and flexibility it provides; it also derives from the way more complex commands are built upon simpler ones, enabling users to start by learning commands for the most essential cursor movements and then, based on what they already know, to learn more and more commands until the full set is at their ready disposal. In the same way, one can go from learning the basic block operations (move and delete) to learning less commonly employed ones (append, print, and save), to learning those that will only be used in rare circumstances (boldface, underline, and shifting the whole block to upper or lower case). WordPerfect may use a great variety of command structures, but the learner will come to recognize that it uses them consistently in relating functions.

WordPerfect is a program that anticipates complicated demands rather than simpler ones. Thus, line spacing is reset by the "Set Format" command, which allows adjustments in margins, tabs, "extended tabs," and the "hyphenation zone" at the same time. It is also a careful, precise program, as illustrated by this explanation of its "window-orphan protection" feature:

Widow-Orphan Protection

A widow refers to a first line of a paragraph appearing by itself on the last line of the page. An orphan refers to a last line of a paragraph appearing by itself on the top of a page.

Widow/Orphan Protection

To avoid widows and orphans, press the **Set** key and select **Widow/Orphan**. The current setting is displayed, and you can press **y** to change to yes, and **n** to change to no. If you change to yes, then from that point in the document forward you will avoid most widows or orphans.

You may turn widow/orphan protect on and off as many times in the document as you wish. One line subject headings and paragraphs of three lines or less are not protected, so you may want to use the **Conditional End of Page** in these cases.

All WordPerfect's many options do not come without some costs beyond the monetary. For one, the manual contains thirty-seven pages of densely packed installation instructions, which must be closely followed if all the program's features are to work. For another, the user must adjust to working with over fifty function abbreviations that do not normally appear on the screen. These include those for operations already enacted on the display (like [C]...[c] to indicate the beginning and end of centering, as well as those that are to be enacted in printing (like [Adv △] to indicate the printer is to move up half a line). There is a "Reveal Functions" command to put these symbols on the screen along with the text, but unfortunately it will only do so for three lines above and three lines below the cursor, a limitation that makes checking and correcting these embedded codes for any substantial block of text

difficult. Also, in suppressing all these indicators, it suppresses one that most users would prefer always to have on the screen: the end-of-paragraph indicator.

Another (small) flaw in the program design is that the user who wants to edit an old file cannot do so immediately on entering the program, since a blank file screen (ready to accept a new document) always appears first. The new user may find, as I did, that the "Rewrite Screen" command (Shift F1) does not do what its name implies--and what it does do is much more extensive reformatting than wanted, necessitating a tedious series of maneuvers to redo the file into the desired shape. The program does come with the "spell/edit" sub-program Speller, which can be accessed without exiting your text, which has a 1500-word list that resides in the computer's active memory (for speedier checking), and which permits up to 10,000 words to be added to its dictionary--but which only contains 30,000 words, thereby limiting its usefulness significantly. At times, WordPerfect operations can seem bewilderingly like a series of Chinese boxes, one inside the other--see, for instance, directions on "Changing a Primary File within a Merge." Those who want a split-screen capability or need double-column printing will have to look elsewhere, as will those who want a program that will create tables of contents and indices.

Most users, however, will find that WordPerfect's weaknesses are greatly outweighed by its strengths. It is conveniently designed with insert and right justification as defaults, has a clear page break symbol (a broken line), and spaces text on the screen the way it will appear in print. It is generally good too about asking the user to confirm large-scale replacements before proceeding with them. It will do horizontal scrolling, a line or screen at a time, center a page from top to bottom, provide vertical lines to create boxes (when combined with underscoring), do foreign language and Greek alphabet characters and a variety of mathematical symbols, and print multiple copies. It is a quick program in operation, and even comes with a faster ("Flash") version for the hard-disk IBM-XT.

WordPerfect's on-page footnoting capability is extraordinary. Not only is creating footnotes simple, but the renumbering and repaginating of them is automatically done as others are added or deleted. The program does not, however, offer the option of collecting footnotes together at the end of a text (although one could presumably do this "by hand," i.e., moving each of the in-text footnote numbers to the end of a document after it was completed).

WordPerfect is also very strong on creating form letters and performing allied functions, like printing address labels. Like Spellbinder, it has "macros," though they are not nearly as extensive or powerful as that program's. Since WordPerfect does not come with any prewritten macros, all of them must be created and saved by the user. Essentially macros are used for storing repeated functions, and so form part of the program's overall mailmerging capabilities. They can be used in several ways: for storing and retrieving "keyphrases" for insertion into a document, for creating conditional commands, and for embedding functions and formats within one another. Their most immediate use will probably be for storing standard print formats so they can be invoked (macros are never "retrieved") with a few keystrokes. They will also perform functions like removing all underlinings and centerings from a file. Although the manual never quite says so,

apparently macros can contain any combination of characters and functions--so their usefulness is likely to increase over time, as an operator discovers more and more formats and functions storing for easy "invocation." Many new users will find, to their relief, that macros are not something they need to bother with early on--or even later on for some kinds of work. In an office setting, however, new users may have to take the time to learn their creation and operation in order to complete some routine tasks, a procedure that will slow the initial learning process, but one that will insure that high-quality, intricately formatted and precisely individualized documents can be created quickly and accurately.

WordPerfect also includes a somewhat misleadingly named "math package," which deals, in complex and useful ways, with arithmetical functions. It will, in the manual's words, "let you add, subtract, multiply, and divide across columns, and calculate subtotals, and grand totals down columns." The real utility of this feature is indicated by the series of increasingly complex exercises the user is given to learn its operation. These include the creation of an invoice, budget, and financial statement, and culminate in the creation of a quarterly stock report.

WordPerfect may have been created with chiefly business situations in mind, but it is also a program that will function very well in most academic situations. Because it is designed more for long-term efficiency and flexibility than short-term accessibility, it would not be an especially good choice, though far from the worst, for students who will be using word processing on a limited basis or over a brief period. However, for students who will use word processing extensively and variably, this is a program that amply rewards the money and learning time spent on it. Many faculty also, I think, would want to consider WordPerfect as their word processing program, especially if they plan on doing extensive writing or rewriting of complicated documents right at the terminal, since this program is, in the main, screen-oriented and still very powerful in its editing and formatting features. (As a general rule, the more "original" composition done on the terminal, the more useful it is to work with a screen-oriented program.) WordPerfect is, though, a program only faculty who are sure they want to do extensive work with word processing should investigate, since its relative inaccessibility at the outset is likely to discourage the uncertain.

As noted earlier, if one tabulates the number and variety of functions a word processing program can do, then Perfect Writer is probably the most "complete" program available--but if one also includes certain desirable design features, most particularly screen orientation, in the equation, then WordPerfect would probably be the most "complete." The quotation marks around "complete," however, indicate that, the denomination of these two programs aside, no single program can be said to be "perfect" for all users in all situations.

MICROSOFT WORD

Distributor: Microsoft Corporation
10700 Northup Way
Bellevue, WA 98004
206-828-8089

Cost: $375

In one important respect, Microsoft Word is the antithesis of programs, like WordPerfect and PeachText, which have little but text on the screen. Word presents a very complete and complex (but not really complicated) editing screen, divided into a "text area" above (with a double border around it) and a "command area" below. The former may contain up to thirteen different elements, including ruler line, window number, style bar (a vertical column for indicating style symbols), end mark, and footnote window. The latter may have up to eight elements, document name, status line, scrap area (for storage of the most recently deleted text), and help mark (for requesting help files) among them. Operators who prefer to work with a very full screen display, one that provides them with a great deal of information about their text and the command options available to them, will find that Word provides such in very full measure. Consequently, the program is almost as "screen-evident" as VisiWord, and if more complex, it is also considerably more powerful.

Word will also, apparently, be the first full-featured word processing program to be made available for Apple's very popular new Macintosh computer, which instead of command keystrokes uses a "mouse"--a hand-held rectangular box, attached to the computer by a cord--to move the cursor around the screen (as the mouse is moved across a flat surface adjacent to the terminal). For the time being, then, Macintosh owners who want a sophisticated word processing program for their computer will have to opt for, and adapt to, Word--or go without. For most, the process should not be too difficult.

Since Word is available for use with a mouse on the IBM personal computer as well, I tested the program using one, rather than by just using the full alternate set of keystroke cursor movements. What using the mouse provides, for those who can adjust to using it (a few probably cannot), is a much quicker way of moving the cursor (which Word, incidentally, does not call such) around the screen. For example, moving up ten lines and left nine characters to make an insertion might take twelve or fourteen or even twenty keystrokes in most programs, but using the mouse makes it merely a matter of moving one's hand ten inches in one direction and four in another, then "clicking in" the new cursor position by pressing a button on the mouse. The only problem, at first, will be learning to move the pointer, the arrow that responds to movements of the mouse, accurately enough to land exactly on the desired space instead of having to perform a series of small adjustments to get into the space.

Word's primary design is based on its use with the mouse, which results in some idiosyncracies when it is compared with older programs. For example, instead of speaking of "cursor movements," the manual speaks of ways of

"selecting text," thereby making keystroke movements sound rather like ways of using the mouse to move over and choose text.

Another noticeable aspect of Word's design is its organization by classification and division, which resembles the organization of VisiWord, although Word's organization is at once more complex and less rigid. Below the text area on the editing screen, seventeen alphabetically arranged commands control most of the larger program operations, which the user gets to from the editing screen by pressing Esc. Many of these commands, like Copy and Print, are obvious by their designation, but a few, like Jump and Gallery, refer to Word's particular way of going about things. One command, Alpha, which puts you back in the editing screen from the command sequence, is not at all obvious and rather difficult to recall at first. (Using Esc as a toggle would have made more sense.)

Most of Word's interrelated commands are effectively designed, with more complicated and less frequently used commands built on the more basic commands. Thus, Return starts a new paragraph, Shift-Return a new line, Ctrl-Return a new division, and Ctrl-Shift-Return a new page. The keystroke ways of "selecting text" (i.e., moving the cursor to it) are especially logical and well-related: Home moves to the first character in a line, End to the last, while Ctrl-Home and Ctrl-End move to the top and bottom of the screen; F1 moves to the next window of text, Shift-F1 to the previous one. Together with other cursor movements, these make for such a complete, easily remembered and employed set that many users will normally find themselves only using the mouse to make certain long-range movements or movements that involve several directions.

The command sequence is supplemented by five toggle commands: Scroll Lock, Number Lock, and Capital Lock, which use the IBM keys so designated, and Overstrike and Extend (which repeats an operation, like boldfacing as the cursor is moved over words), which use F5 and F6.

Word also has a set of commands that work through the Alt key. Alt followed by a number immediately shifts the line spacing to the designated number, and it reflects that spacing on the screen. (Unfortunately though, it does not directly permit half-line spacing, although such spacing can be approximated by resetting the "pitch" per vertical line). It also has an Alt 0 option for writing single-spaced text with a blank line between paragraphs: each paragraph return skips the cursor down two lines. Most enhancements are also done via the Alt key and are designed simply and mnemonically: b for boldfacing, c for centering, u for underlining, + for superscript, - for subscript, j for justification. It also offers two other character enhancements: d for double underlining and k for small capital letters. All of these enhancements are enacted on the screen.

Word's set of deletion operations is not extensive--there are no specific commands for deleting a word, sentence, or line--but its backup feature on deletions is excellent. For correcting typos and deleting a few characters, there is the backspace-delete key or the overstrike function. There is also a method for deleting by paragraphs. In the case of most medium and large-scale deletions, the text must be marked and then removed via the Del key, which puts the deleted text in "scrap" from which it can be retrieved until the next deletion replaces it there. (The unit currently in scrap is indicated at the

bottom of the screen.) For deletions about which the writer is sure, there is Shift-Del, which simply deletes without putting the word(s) in scrap, a useful command when one wants to keep a certain dubious passage available in scrap for possible reinsertion while making subsequent deletions. Word is also very good about backup on a large scale: namely, it automatically creates backup files.

One of the things that permits Word to include so many details on the screen display is that text is very clearly set off from other material by a double-lined "window" around it. When more than one text is on the screen--Word permits not just split screens, but up to eight windows--each is clearly set off from the others by its border. (Although putting eight texts on the screen at once would mean only a miniscule, and probably insufficient, part of each would appear, the ability to have three or even four texts on screen opens up some intriguing pedagogical possibilities.) Besides text windows, Word also has a footnote window, a very useful feature when one wants to judge the phrasing of a footnote against the textual reference to which it refers. (Footnotes are written, and automatically numbered, via "Format Footnote" operation.) Although Word may not be the most thoroughly screen-oriented program, it is certainly the one that allows the operator to see the widest variety of information on the screen display while still showing text more or less as it will look when printed.

Like other well-designed programs, Word has some shortcuts that experienced users will appreciate. For instance, a file can be called up as part of the command to call up the program itself. F4 tells the program to repeat the last command given. And it is possible to send keystrokes directly to the printer for typing things like envelopes. There is also the option of getting a "draft print," which can be printed much faster (since it does not stop to do character enhancements), when one does not yet need a fully finished copy of a document. In addition, Word is set to remember from one session to the next whether most options are off or on. Thus, users who prefer, say, overtyping on and having the normally invisible screen symbols made visible, but do not want line breaks shown "as if printed" or the warning alarm to beep, can have the system always operate that way without first having to go through a series of adjustments at each session.

Needless to say, as full a program as Word is, it can, more than occasionally, confuse new learners. It can be perplexing to encounter a "Quick Reference Guide" (a three-page cardboard foldout) that contains nearly 150 commands, sometimes bewilderingly in tandem: e.g., "The following symbols will appear in the style bar when you set the style bar option with the Window Options command." Conversely, as operators become experienced with Word's operation, they may find the program's very full screen display, which contains much information they have already assimilated, rather a distraction from the text they are entering on that screen. The fullness of Word's screen display also means that only twenty-two, as opposed to the usual twenty-four, lines of text appear on the screen at once. Some users may find that while Word is basically screen-oriented, it is not screen-oriented enough for them. For instance, it allows text to be created in two (or more) columns, but it only displays one column of text at a time. Other users may find that Word should be a bit less screen-oriented and display a few more format symbols, particularly the end-of-paragraph symbol in all cases, on the screen. (One

can get desired symbols displayed, but only as part of an option to display all the screen symbols, which makes for a cluttered screen.)

Word is, more or less, a "straight word processor," if a very powerful one. Thus, users will not find in it certain auxiliary functions that various other programs sometimes provide, like the capacity to do mathematical calculations. Although it can do mailmerging tasks, it does not have the full mailmerging capability that some offices might need. It cannot match programs like WordPerfect and, especially, Spellbinder in their capacity to store complex operations as easily invoked "macros." (It does, however, store formats.) Nor does it match the editing flexibility of programs like WordStar or SuperWriter, although it does offer some odd and perhaps useful editing commands (like those to move right or left 1/3 of a line) that they do not. It also lacks the capacity to create tables of contents and indices, capabilities one finds in a print-oriented programs like Perfect Writer. For certain uncommon types of documents, Word's limitation on line length--the smaller of 254 characters or 20"--may prove a prohibition. And while it may just be my personal reaction, other users too may find Word a bit "characterless"--i.e., it presents no overwhelming paradigm for the way it organizes and goes about tasks--and so is just a bit harder to learn thereby. Still and all, Word is clearly one of the very best of the newer programs on the market--powerful, efficient, clear, and well-designed.

Word's documentation very much mirrors the program it describes. It is sophisticated, clearly written, and thorough. Unlike the screen display though, it seems to rely heavily on technical terms to explain program operations. (It assumes a reader already familiar with word processing and computer equipment.) It makes few concessions to the learner who just wants to understand basic operations and get on with using the program. Instead it explains the many elements of the program in full detail, rather than explaining them briefly at first and more extensively later on. Also, instead of a single reference chapter, it has two: "Elements of Microsoft Word" (e.g., "Selecting Text," "Requesting Help") and "Command Directory" (e.g., "Library," "Transfer"). Its "Quick Reference Guide," as already noted, is likewise fulsome, and of more use when doing complex documents than when just learning the program. Because of its documentation, Word is a program one learns slowly but thoroughly.

Several features of the documentation do aid in learning the program. Unlike WordPerfect and a few other programs, when Word renames keys, it makes clear to which keys on the IBM keyboard these names refer. Its "Keymap," which appears both in the manual and on the "Quick Reference Guide," has three columns for entries--Action/Keytop/Name--so the user can always see what key(s) correspond to the program names for them. Similarly, the manual's explanations include frequent allusions to the "keytop" designations of the commands it "names," so the reader need not keep turning pages to discover what keys a command entails.

In line with its sophisticated, thorough approach, the manual gives the learner some rather difficult and complex tasks to perform--these may take some time and cause a bit of frustration in their doing, but they insure a high degree of proficiency for those who complete them. An additional help to the learner (and even the experienced user) is the very complete message directory at the back of the manual. Clearly, Word's documentation is

designed to make it a program that is learned once and completely, rather than in several increasingly complex and extensive stages.

Word is certainly a program to be recommended for many student and faculty uses. With students new to word processing, instructors would need to analyze the program thoroughly and devise an effective "Quick Start" course so students could begin producing documents after just a few sessions at the computer. Word might be too complex a program for students who will only be using word processing simply and briefly--say, for one course for one term--but for that growing percentage of students who need a word processing program that will serve them over the course of their college career, Word will more than repay the time invested in learning it.

Like another of the newer programs, Volkswriter, Word has a subprogram just for the conversion of WordStar files to its own system--and like Volkswriter it is a program that will tempt a number of WordStar users (and those using other older programs) to switch over to it. For faculty new to word processing, Word offers a particularly attractive combination of features: it is basically screen-oriented; it has a very full screen display to prompt and guide the user; and it has considerable power and flexibility for those working in the English language. (They will, however, be somewhat put off by the rather technical approach of the manual.) Those in mathematics, the sciences, and foreign languages will find a number of other programs more obviously suited to their needs. What Word lacks, like the capacity to create and store macros containing specialized sets of directions, are things many faculty might not need. What it conspicuously has, like the ability to put footnotes in their own window on screen and print them at the foot of the page or collect them together at the end of a document, might very well prove to be assets many faculty members need or would like to have.

Unlike the many sophisticated programs that seem to have been designed primarily with an office setting in mind, Word seems conceived more with the "general" writing situation in mind. Once they learn the program fully, many student and academic writers should find Word much to their liking; the rest should be able to tell fairly early on that Word is not the word processor for them.

POWERTEXT

Distributor: Beaman Porter, Inc.
Pleasant Ridge Road
Harrison, NY 10528
914-967-3504

Cost: $399

PowerText, as its name implies, is a big, strong program; at times it also seems, to carry the bodily analogy further, musclebound. For one thing, its manual is a large (10-1/2" X 12"), heavy (5 lbs.), unwieldy grey looseleaf

binder that will fit next to some terminals only clumsily, and next to others not at all--in either case, making the frequent reference to it which the program necessitates very difficult, if not impossible. For another thing, this very complicated print-oriented program makes even fewer concessions to screen orientation than, apparently, it could. (It is the most thoroughly print-oriented program analyzed here.) Still, PowerText offers such a range of print features that anyone working in the English language who needs a program to do very complicated documents or to do several very specific kinds of documents repeatedly should consider PowerText. All students and almost all faculty, except those involved with the publishing of scholarly journals or like works, can safely disregard this program. It might, however, serve very well in some college and university offices, like the admissions office or the grants office, where form documents or irregularly formatted documents are part of the normal workload.

In the considerable (at times impressive) additional documentation and sample documents that accompany the PowerText manual, the program's designers stress that their program is unique in emulating not the electronic typewriter, as other programs reputedly do, but the secretary:

> Someday it will all be different. From your office, car or maybe even your back porch, you will dictate into a microphone. The computer at the other end will recognize your words. Since it will know what a letter looks like, the way you want it typed, it will perform the transformation, and when it is finished, if the letter is going to someone who cannot receive electronic mail, it will type the envelope. If you have dictated a report, the system will create a table of contents for you, and perhaps a cover page. It may even switch to little Roman numerals for you.
>
> The PowerText system has been designed as a major step in this inevitable evolution.

At the heart of the program's design, then, is an attempt to anticipate the future of word processing and secretarial work, when the development of voice-activated word processors should markedly alter the usual business division between the writer (executive) and the producer (secretary) of texts.

As the program now operates, this claim is, at best, only a half-truth. With PowerText one cannot simply begin "dictating" text, but must first perform a rather complex series of operations, like "setting environment" features. For example, most users will want justified right margins and so will need to change the "filling" from "false" to "true" before they write. Many new users will be confused by the information they receive and the operations they must perform before going to the editing screen. The bottom of the "Set Environment" screen might tell them something like this, information that will be meaningless until they become quite experienced with the program:

```
4 bytes used, 32251 available.
There are 0 pages in the left stack, and 0 pages in the right stack.
You have 939 pages of room, and at most 1 pages worth in the buffer.
```

To discover what all this means, they will have to wade through detailed, copious, and not particularly accessible descriptions in the manual. (PowerText's manual is as technical in content as it is large in size.) When they do get to entering text, they will almost immediately have to start using a series of embedded commands, to do centering, spacing, underlining, and other common functions, and they will have to use them without any indication on the screen as to whether the commands have been correctly entered. (Even centering is not enacted on PowerText's screen.) The program may emulate the electronic secretary, but emulation is not performance: the user must still do a great deal of the work.

PowerText is a program adapted to, not written for, the IBM personal computer. Thus, it makes no use of the ten function keys (although it does allow them to be programmed). Instead, it relies very extensively on embedded commands--indicated by a slash in the first space on a line--to signal many of its operations (to be enacted at the print stage, of course). For example, "/spa [2]" gives double-spacing, "/cen" centers a line, and "/inc" followed by a disk name and file name includes another file in the one being printed. A supplementary set of commands uses the caret as a toggle: "^b...^" boldfaces, "^d...^" does subscript, "^u...^" does superscript and "^f...|" creates a footnote (which is printed within the body of the screen text, directly after its reference, and later extracted to the bottom of the page).

PowerText's footnote feature, which is powerful and flexible, is nonetheless a typical instance of how the program makes little concession to screen orientation. Almost no user would prefer or find it clearer to have footnotes appear in the middle of the screen text--since they easily could, as other programs demonstrate, be put in a separate file of their own or, even better, be put in a window at the bottom of the page. In fact, very few of PowerText's on-screen indicators will help the user envision what the text being created will look like when it is printed out, the page break indicator, a line of double hyphens with "END OF PAGE" in the middle, being the most notable exception. (Unlike several other print-oriented programs, PowerText does not wait until the file is being prepared for printing to divide it into pages.)

PowerText is another of those programs where, more unfortunately than not, the user must always be in one mode or another and must switch back and forth between them quite often. Thus, the prompt line at the top of the screen when one edits (or creates) a file is this, initially confusing, string:

A(djust C(opy D(elete F(ind I(nsert J(ump R(eplace Q(uit eX(change ?

When creating a new file, one does not even choose one of these to begin, but rather presses "S(et," then selects "E(nvironment" to establish margins, page length, etc. When this is done, "I(nsert" is chosen from the original prompt line, since PowerText defines the entry of any text--whether new text at the end of a document or words added to already written text--as an insertion. The I(nsert prompt line looks equally confusing:

Insert: Text {⟨bs⟩ a char, ⟨del⟩ a line} [⟨etx⟩ accepts, ⟨esc⟩ escapes]

One thing this line indicates is that pressing Esc will discard all the text that has been entered to that point--rather too easy a way to make such a large removal. Etx, the End key, will save the text just as simply--but it would seem more logical to make removing a whole file more difficult than saving it, since text accidentally saved can always be discarded, but discarded text can be permanently lost.

If PowerText is based on any model, that model might not be other word processing programs, but rather other kinds of computer programs, particularly ones that rely on a great variety of symbols and abbreviations to denote their operations--and so have a very "technical" look to them. PowerText's command structure and screen display do not suggest that one is working with a continuous piece of writing in English so much as with an assemblage of material (text) on which a series of operations are to be performed. For some operators, particularly those entering text composed by someone else, this "distancing" quality may prove something of a benefit, because it separates the abbreviated, specialized command use of language from the fluid, more continuous use of language in the document proper. PowerText's single greatest advantage--its six prewritten "format files"--also will prove most beneficial in a secretarial, as opposed to a composing, situation. In the manual's words,

> The format files are named: DOCUMENT, LETTER, PERSONAL, LANDSCAPE, STANDARD and MEMO. DOCUMENT specifies the format of a paper, a market report, a business proposal etc. LETTER provides a style for standard business letters. PERSONAL describes a personal letter format. MEMO describes an inter office memo style. LANDSCAPE defines a wide page with 132 character margins, and STANDARD describes a blank page without any headers or footers.
>
> The supplied format files can be changed to reflect the personality of your business style and they can be augmented with other format files created to suit your needs.

Once you learn to use these formats, you can use the "style utility" to create additional format files. Thus, PowerText can, from the beginning of its employment, provide a variety of highly structured, commonly used formats into which the user can simply enter text--and it will provide a means of creating additional formats once users become experienced with the program. Since the program is good at anticipating just the kind of documents that will be produced repeatedly in a business office, its greatest utility is likely to be there.

Any writer regularly concerned with producing documents which must look as good as possible will appreciate a number of PowerText's features. For example, the program analyzes the whole document before deciding where to put the page breaks. It is very good about permitting localized changes, providing, for instance, several embedded commands that can be used to change margins for a series of lines or a paragraph. Its "find and replace" command is, unlike those in most other programs, case sensitive--thus, you could search out instances of "Bill" without being interrupted by instances of

"bill"--or vice-versa. Some of PowerText's commands are designed, efficiently, to serve a dual function; e.g., there is a command that allows a designated number of lines to be skipped without all the empty lines having to be put on the screen--and entering text in the "middle" of this command can be a means of putting notations on the screen that are not meant to be printed.

Once the user gets used to certain peculiarities of the program, PowerText's command design becomes more understandable and flexible. Esc is a quick way of getting rid of a whole unacceptable file in one keystroke (as well as a too-easy way for novices to lose a file). Saving files as one goes along is simple enough: pressing "Q(uit," then "B(ackup re-edit same file." Seeing footnotes right in the middle of screen text will continue to bother most operators, but it does provide the advantage of viewing footnotes in relation to the text they reference. Moreover, the program can be directed to collect footnotes together at the end of the document, as well as to put them at the bottom of pages.

Despite its heavy print orientation, PowerText has some features that definitely do make editing text easier. The most recent deletion is removed to a buffer, from which it can easily be retrieved--so the threat of accidentally wiping out large sections of a file via Esc is less worrisome. At certain points the screen display is very explicit about distinctions in operations--for instance, the first three options on the print menu have a commendable thoroughness in their wording:

> D(isplay your file on the screen.
> P(rint your file on continuous form paper.
> T(ype your file on individual sheets of paper.

As the first of these options indicates, PowerText does make some provision for its lack of screen orientation by allowing the user to preview text on the screen as it will look when printed. And although it does not use the ten function keys on the IBM keyboard, it does allow them to be programmed, a feature some experienced word processors will find a helpful way of providing shortcuts for certain maneuvers they regularly perform.

PowerText's range of print features will seem extraordinary to some users and merely bewildering to others, the difference probably being whether the user needs regularly to employ a fair number of them. It has a "newspaper command" to print in double columns, a "table of contents command," a command just for creating glossaries, another for making outlines, and even a subprogram, PowerScript, for writing screenplays. On a smaller scale there is a "no-edit" command for printing text exactly as typed in (so special spacing effects are not regularized), a "box" command that will automatically put a box around a designated section of text, and a command for creating horizontal borders (of stars or equal signs or whatever other symbol is indicated). Twenty-six strings of characters (each up to eighty characters long) can be created and inserted in text via an embedded command. Another embedded command will signal the program to treat characters following it as hexadecimal. The program is also "smart" in a number of respects. In formatting, items can be entered in random order and PowerText will sort them out. The program even has a "K(runch" command which takes unused blocks on a disk and puts them together at the end, so the user can get the maximum usage of disk space.

As several times indicated, almost all the features of PowerText come at the expense of screen orientation and require working with a screen very full of command strings. Moreover, that screen also lacks a status line and other help indicators, so the learner (or even the veteran) has only the manual to go to for reference at some points, and nothing to go to at others. The user must also constantly remember to switch in and out of the "modes" and not to get stuck "between" them, a habit at least some users will find difficult to acquire.

Several of PowerText's design features are poorly done. If one neglects to name a file, the program automatically designates it "SYSTEM.WRK." And if one forgets twice in a row to name a file, the first file could easily be lost since the second would be given the same "SYSTEM.WRK" designation and so would replace the first when it was saved. Using the hyphen symbol rather than the underline symbol (on the same key) for underlining seems a needlessly perplexing substitution, as is the use of # to signal paragraphs in a footnote. (Footnote paragraphs do not appear as such on the screen.) Some of the program's phrasing is obscure--e.g., "C(hange name of destination file" is the operation one chooses to save an edited version of a file under a new name so that the unedited version can also be preserved.

For all its power and flexibility, PowerText has a few surprising lacks. It will not do soft hyphens, and it can not vary spacing within a single paragraph. It cannot create foreign language characters, has no spooling capacity, nor a true split-screen capability for working with more than one file at a time.

The PowerText manual is decidedly not a great asset either. As bulky in content as in form, it begins with eight long, very full "lesson" sections, which thrust the new user into full and complex program operations from the first. (PowerText could use a good "Quick Start" course.) The following chapters, via alphabetically arranged entries, deal with embedded commands, editing functions and concepts, and format files. If still somewhat overwhelming, these chapters are at least clearer and more accessible than the lessons. The manual's wording sometimes sounds very peculiar, combining near jargon with odd word choices:

> Because the text file itself is modified by the actions of the file buffer, a separate copy of the file is created at the start of an edit session; this provides the ability to restore the original file contents when an edit session is exited or interrupted by dire circumstances. The copied file is called a backup file.

As for supplementary material, the manual has an excellent section on a variety of printers (a pageful of information on each) and an absolutely useless "Command Summary" (which omits mention of the keys used to enact the commands). Its sample documents (several of which are not included in the manual per se) are impressive demonstrations of what the program can do in formatting and printing English language documents.

PowerText can be had with PowerSpell, which will fit on the same disk as the main program and so be accessed from it, and PowerNet, which provides for interface with other systems and other users. Unlike most program

manufacturers, who sell their program disk "as is" (the reason it is so important to make a backup copy immediately), Beaman Porter will replace a faculty PowerText disk for up to five years.

Although the company is working on a simpler program, PowerPen, specifically designed for student use (on the Apple IIe and IBM PC Jr.), this new program is not likely to be the best one for students if it is modelled on PowerText in any major ways. Certainly, PowerText's assets are not those that students particularly need, and its drawbacks, particularly its very complex embedded command structure and consequent lack of screen orientation, are things that would make it very difficult to use in a pedagogical situation. PowerText was designed for the office and is best and most properly and profitably kept there.

"Complex typing is trivial with PowerText," and "Creating a new format file is [apparently likewise] trivial," boasts the long sales letter accompanying the program. Unknowingly, the program's producers have indicated through their misuse of the word, just what might be wrong in a large way, not just with their (in many ways good) program but with word processing programs in general: namely, that they distract the user with such a multitude of editing and formatting options that what the user is actually writing can be overlooked and can indeed become "trivial."

MULTIMATE

Distributor: Multimate International Corporation
52 Oakland Avenue, North
East Hartford, CT 06108
203-522-2116

Cost: $495

By itself, MultiMate may be the single most expensive program analyzed here (although it includes several sub-programs that one usually must buy as peripherals with other programs). To wit, MultiMate requires 256K to run, so the user may have to purchase "extra memory" to enable the program to run on his or her system. For example, the IBM-PC comes with 128K, and an additional 128K costs about $180. Of course, institutional computers may already have increased memory capacity to run other kinds of programs, or the personal user may need to increase the computer's memory to run a number of non-word processing programs anyway. However, not many individual users who could not get some sort of discount would want to pay the nearly $700 it would cost to have MultiMate if they intended only to use their computer as a word processor.

With educational institutions it is a different matter. Since MultiMate can be had at a very handsome educational discount, and since it is more than likely that institutional computers have or will eventually need increased memory capacity, Multimate would be one good choice for use in colleges and

universities. It is sufficiently screen-oriented and easy enough to learn, yet powerful enough in editing and formatting features to make it attractive to students who will be using it for a variety of writing tasks over their years in college. It also would be a suitable choice for use in most college offices, since it has clearly been designed with "business" situations in mind.

With the possible exception of its documentation (see below), MultiMate does not surpass the other expensive programs reviewed here in any single respect. It lacks the editing flexibility of WordStar, the range of print options of PowerText, and "macro" capacity of Spellbinder. It is not as easy to operate on screen (as "screen-evident") as VisiWord, nor as thoroughly screen-oriented as EasyWriter II. It does, however, do a fair job of approximating all these qualities. MultiMate's greatest strength is that it is probably the best balanced program in its class--it is not extremely strong in some respects and very weak in others, but "fairly strong" in almost all ways.

MultiMate's manual avows that "although menu-driven, the system is not menu-cumbersome," an assertion that is truer than not. Its screen is basically clear and usually no more complex than it need be, which is somewhat complex since this is a powerful program. Rather than speaking of "screen orientation," the manual speaks of "page orientation"--and the program does, indeed, often work in terms of page units. For instance, each time a page is completed--pages can be up to 150 lines, with 55 the default--that page of text is automatically saved to disk.

Moving the cursor around on the screen is easily done and logically arranged: using Ctrl with the left or right arrow moves the cursor a word rather than a character at a time, and using Alt with the arrow keys scrolls left or right. The nontext part of the screen is well-designed, compact, and includes a space to display "Invalid Key Messages" (like one telling the user that the Del key cannot be used in the insert mode).

MultiMate is another of those programs which does not hesitate to involve the Ctrl, Alt, Shift, and ten function keys in its command sequences--and while it does not always employ them in the most consistent or logical way, the program does provide excellent help, both on the screen and off, to make individual commands easy to locate and enact. MultiMate's eight commands for controlling format lines are a typical instance of the program's only half-consistent use of command keys and its rather vague way of interrelating commands. Four of the eight use F9 (by itself, with Shift, Alt, and Ctrl), two use F6 (by itself and with Shift), and two simply use letters (D to delete, F to "setup"). This group may be difficult to remember, but when the whole list can be called onto the screen in three keystrokes, the operator need not worry overmuch about memorizing it.

Enhanced characters are created by a series of Alt key commands that are, again, not mneomonically arranged. Alt Z is boldfacing, Alt L creates the British pound symbol, and Alt W-Alt Q gives subscripting (with the commands reversed for superscripting). As with WordStar, these commands are toggles and are not enacted on the screen, but instead put unprinted designations on the screen to show where enhancement begins and ends (a solid rectangle with

irregular edges for boldfacing, up and down arrows for subscripting and superscripting, etc.).

On the other hand, MultiMate does use some keys straightforwardly: Esc interrupts operations (and is not involved in any command sequences); Home and End move the cursor to the first or last character on the screen. Somewhat similarly, MultiMate uses the plus and minus keys on the numeric keypad for character insertions and deletions. (The program has, rather unhelpfully, strikeover as the default.) Larger deletions are done by highlighting text and pressing Del.

MultiMate is thoroughly organized "up front"--not only do you type in the file name before beginning a new file, but you are also given an optional "Document Summary Sheet" to fill out for it. This includes single-line entries for the file name, author, addressee, and operator, and five-line spaces for "Identification key words" and "Comments." As the document is created, the document summary sheet reflects the total number of pages and keystrokes. When it is edited, the user can choose to enter a "Modification date" and even get a reading of "Keystrokes last session."

Because MultiMate is a more rational program than not, most users will adjust to it quickly. Its main menu, for instance, makes this important distinction in its first two choices: "1) Edit an Old Document 2) Create a New Document." Thus, new users will not be confused about having to "edit" documents they are yet to create. The user is always kept informed, via the status line, exactly where the cursor is in a text. Since MultiMate allows for the easy storage of formats, it is a simple matter to invoke one at the outset of a file, or elsewhere in it. Several functions regarding the whole text or designated portions of it--like page numbering, alphanumeric underlining, and continuous underlining--can be made automatic. And while many of the commands are not mnemonic, a number of the more important ones are; e.g., underlining is done as on a typewriter (the shifted hyphen/underline key), and it is represented on the screen.

MultiMate has almost all the "usual" advanced features one expects to find in an expensive program, like the capability to do soft hyphens, headers and footers, and create forced page breaks. Its "go to" command works, very usefully, by page designation, allowing you to get to any point in a long file quickly. It has a "pause print" command that is embedded in the text, with a clear symbol (a hollow box with a pitched "roof") to signal its location. When working with columns, the user can press Shift F3, which puts the program in the "column mode" and offers a further series of commands just for working with column text.

MultiMate has several good backup features. As already noted, it saves each page to disk as it is completed. In addition, a page can be saved to disk at any point in its creation simply by pressing Shift F10. Among its four "Utilities," sub-programs that run out of the Disk Operating System, is "RECOVERY," which allows large accidental deletions, including whole files, to be recovered. RECOVERY also allows files to be recovered in case of disk error, a feature any user who has had a file lost because it was on a "bad sector" will much appreciate. Still, the user will want to put backup copies of important files on a second disk, since MultiMate does not automatically

create its own backup files. It also, unfortunately, lacks a delete buffer from which small and medium-sized deletions can be recovered.

One of MultiMate's definite assets is its "key procedures," a feature that allows any combination of keystrokes and pauses between them to be permanently stored and invoked with a few keystrokes. Thus, one could create a key procedure for a form letter or standard report, with a pause at each point where information was to be typed in from the console--a more efficient way of creating such documents than having to type separate lists of information to be merged into the standardized form. While not as powerful or flexible as the "macros" found in some programs, MultiMate's key procedures are considerably easier to create and employ.

MultiMate is very thorough in setting up a document for printing. When a document is "submitted" for printing, the operator is given a long series of options, such as getting a draft print (single struck) or a finished print (double struck), having the right margin of the document justified, printing more than one copy, printing the "Document Summary Sheet" along with the document, and "printing in background" so a second document can be edited during the printing of the first. (Naturally, all the options have default settings so the user can bypass any that need not be changed--or accept all the defaults and go directly to printing.) Rather amazingly--Is this one of the reasons the program needs 256K?--MultiMate allows up to thirty documents to be queued for printing one right after another, and it will provide a list of the documents in the queue at any point and permit their rearrangement.

MultiMate comes with a fairly large spelling checker, though no exact count of the words in its dictionary is given. While there are still some bugs in its operation, it does offer the considerable advantage of permitting the user to "spell/edit" a document from the main menu or to check a portion of a document--from a single word on up--from within the document, thereby facilitating those small-scale checks that writers often want to make while writing, but will not make if they must take the time to exit from and return to their document.

MultiMate's single greatest asset may well be its documentation, which is among the very best that users are likely to encounter. For example, it offers three kinds (and degrees) of training. For those in a great hurry, there is a "Quick Start" course in the manual. For those who first want to get an overview of the program, there is an on-screen tutorial. And for those who want to learn and practice all the major operations thoroughly, there is a full set of training lessons (which come in a foldover pad that sets up neatly beside the computer terminal). All three training devices are clearly written and do explain the operations they cover effectively. The training lessons and the much larger manual (a loose-leaf binder which is likewise designed to be self-standing, but which keeps falling over) are excellent graphically, with sharp, dark print on high-quality paper, and with screen text and sample screens in an effectively contrastive green. Early on in the manual, there is a color-coded keyboard chart, which is yet one more effective device that makes MultiMate's somewhat miscellaneous command structure less forbidding and more memorable.

The heart of the manual is its 183-page reference section. Each entry shows, at the top, the keys involved and the purpose of the function; beneath

it has a column of "Instructions" on the left and "Prompts and Options" on the right. The former indicates exactly what steps are to be taken in exactly what order, and adds notes on any qualifying instances. The latter indicates, and sometimes exemplifies, what will happen at each step, including the prompts that will appear on the screen. The reference page on "Hard Space" is a short example of this excellent, understandable reference format:

HARD SPACE

(Alt and S)

Use this function to insert a mandatory space so that words that must be on one line will not be separated when word wrapping occurs.

Instructions	Prompts & Options
1. Move the cursor to the position where the Hard Space character is to be displayed.	You may want to use Hard Space for dates or names which should not be divided on two lines. See example below:
	Example: January ◊ 1, ◊ 1984
2. Press the Hard Space keys — while holding down Alternate (Alt), press S.	The system displays the Hard Space symbol (◊).
Note: The Hard Space symbol does not print out and does not require another space to separate the characters or words.	

Even when such entries go on for a number of pages--as they do with "Key Procedures" and "Spell/Edit"--it is always easy to follow procedures and to see exactly where you are or should be at any given point.

MultiMate's on-screen documentation is also very good. The help key (Shift F1) followed by a command key (or command keys) gives an explanation of the command. For the user who cannot recall the key(s) involved in a specific command, there are five help menus, listing commands and keystrokes for cursor positioning, editing functions, format line control, printing functions, and miscellaneous functions. And for users who are uncertain just what area the command lies in, there is a sixth menu, which lists all commands (and their keystrokes) in alphabetical order--a screen (actually nearly two full ones) that also presents a useful overview of the program's whole command structure. MultiMate may not be the easiest program to operate, but it is certainly the easiest to learn how to operate at any given point, via its on-screen and printed documentation. (The only thing that gives pause in this respect is

that the latest issue of the program contains a short letter, dated 2 April 1984, which lists seven small errors, and their necessary corrections, in the first chapter of the manual--hopefully, MultiMate's producers will not keep discovering other little errors and omissions they have made in the program's documentation.)

For all its power in some respects--like its ability to queue up to thirty documents for printing--MultiMate is by no means as powerful a program overall as many in its class. It lacks advanced editing features like split screens and advanced print features like the capabilities to do double-column printing and on-page footnotes. It will not create tables of contents nor indices. Nor can it handle documents over 150 pages (which will not be an issue unless one is using a disk large enough, as an 8" disk would be, to hold more than 150 pages). In addition, MultiMate makes some normal operations rather cumbersome. Insert is not the default mode--the user must toggle in and out of it--and the fact that inserting text always clears a large space, pushing text following the cursor to the last line on the screen can be as much distraction for shorter insertions as it can be a help for longer ones. Also, the highlighting of all inserted text can be distracting when the insertion is small and the user need not scrutinize it carefully. Unfortunately, the program has no intermediate delete commands between those for deleting single characters and that for marking and deleting text by blocks; while this design simplifies deletion procedures and makes them a bit easier to learn, it does make the (very common) deletion of words, phrases, and lines a more complicated procedure. Moreover, the backspace-delete key works very slowly with Multimate (the cursor hesitates before jumping back); often using this key also causes the screen text to jump up a line, which may confuse new users into thinking they have deleted a line instead of a character.

While the manual is very clear on most points, it does obscure a few important ones--e.g., it needs to collect together and clarify all the points about the program's "page orientation" and to make explicit just what this means in terms of page breaks. (Right now it is very unclear whether the user who is working with standard-length pages--fifty-five lines--does or does not have to enter page breaks.)

Like a majority of the more expensive programs, MultiMate was designed primarily with business and office use in mind. Still, for faculty who do not require the specific features it lacks (like the capacity to do double-column printing), this is clearly one of the better word processing choices, particularly if they are looking for a program that is basically screen-oriented, that provides good, clear on-screen help with a few keystrokes, that has literate documentation, and that does not clutter the screen with a great deal of potentially distracting information as one is entering text. (Cf. Microsoft Word, whose very full screen display will appeal to other users.) Because MultiMate's documentation is thorough and, for the most part, clear, a fair amount about the program can be learned away from the terminal--a factor that makes it an attractive choice for courses where students need a sophisticated program but only have a limited amount of time at the computer to learn it before they must get on to using it for the writings that is the essential work of their course. When the documentation, both printed and on-screen, can be used in conjunction with substantial practice time at the

computer, the program becomes the most effective self-teaching program in its category.

Still, one must remember that documentation itself only provides an entry, slower or faster depending on its quality, to a program. It forms no essential part of the program's capabilities. Other expensive programs may be harder to learn (considerably harder in a number of cases), but they will provide more editing and formatting options than MultiMate if they are learned completely. One reason to choose MultiMate as the program for new student and faculty users who need a sophisticated program is that most of its functions are so easy to grasp that the operator is likely to learn and so employ more if its functions than he or she is likely to do with other programs (say Perfect Writer or WordStar) that numerically offer more functions and options. After all, a program is only as powerful and flexible as an operator regularly makes it in use. On the other hand, experienced operators, who can more easily adapt to complicated and difficult programs, may well find that MultiMate lacks too many advanced features for them.

While MultiMate may be the best balanced of the expensive programs--superb in no single way, save its documentation, but good in almost all--that quality applies only for work in English. It is unsuitable for writing in foreign languages and has no special provision for making the entry of mathematical equations any easier. But for student and faculty members who will be working in English on a sophisticated document level, MultiMate is a good, safe bet.

4.5 Spelling Checkers

Spelling checkers are of three types: 1) those that are an integral part of the main word processing program, 2) those that are an allied program, and 3) those that are a completely separate program from the word processing program. The latter two types are usually larger than the first--and so more useful. Spelling checkers have several times been alluded to in passing--see, for instance, the remarks on SuperWriter--and for some evaluators they are an inseparable part of word processing. For several reasons, however, I consider them peripheral, very probably important and helpful for some users, but just as probably confusing and time-wasting for others.

First, spelling checkers operate on a very simple and limited basis; they check the spelling of any given word against the thousands of words (about 90,000 in the best programs) in their dictionary, indicate every word that does not match, and give the operator the option of entering a correction. Even the most complete spelling checkers will single out proper names, a fair number of uncommon words, and even correctly spelled words. (Most will also, however, allow words to be added to the dictionary.) Their method of operation is useful for catching chronic misspellings (like "seperate") in documents that are chiefly composed in straightforward prose--but it is nearly useless for documents with many proper names, abbreviations, and much special terminology. As a test, I ran the three-page abstract of a conference paper (which included a twenty-item bibliography with many abbreviations and proper names) through an early version of SpellStar--it identified close to one hundred possible misspellings, only one of which, a letter reversal, was actually incorrect. Assuming that I would have spotted this typo, proofing

the document by "eye" rather than by machine would have been a much, much faster process.

Second, and even more important for many writers, spelling checkers will not catch homonym errors. "To," "too" and "two" are all in the dictionary, as are (the more commonly confused) "there" and "their," and so the checker cannot tell the writer when these words are being misused. For basic writers, many of whom are prone to such homonym errors, this inability is a severe limitation of a checker's overall usefulness.

Third, even the better, more sophisticated programs--which often offer alternate choices for words they identify as possibly misspelled--have certain design limitations. MultiMate's "Speller," and every other spelling checker as far as I can gather, has great difficulty with dashes (a piece of punctuation I favor, as the reader can easily see), since dashes connect two words in a continuous space, causing the checker to read them as a single word. Therefore, MultiMate indicated as a possible misspelling "clumsy--can," and offered these seven, not very like, alternate "spellings": consulting, consultant, counsulates, camouflaging, consultation, consultive, clingstone. (Any reader would immediately see, as the checker could not, the inapplicability of this whole operation.)

Fourth, spelling checkers can consume a fair share of the user's time in entering words to the dictionary, a procedure almost all of them allow--although, unfortunately, EasySpeller II, one of the very best, does not. It may take months or even years before a user can get most of the uncommon words (and abbreviations) he or she utilizes into the dictionary, so they will not be flagged as possible errors and so prolong the checking/correcting process.

Fifth, users will need, as always, to proofread their documents carefully anyway, so as to catch and correct non-spelling errors--therefore, prechecking certain kinds of spelling errors will not greatly diminish their work. In addition, some students tend to think, quite incorrectly and detrimentally, that they have completely proofread their document after they have spell-checked it.

All these limitations noted, it still can be said that spelling checkers are useful if employed selectively. For example, EasySpeller II--which works, very "cleanly," through the main program via Ctrl key commands--has a very helpful "Verify" function that will allow the spelling of a word to be checked from the document in just a few keystrokes. EasySpeller II also comes in a medical version and a legal version, each of which adds a long list of technical words to its (already large) 88,000 basic dictionary list.

Those checkers that do offer a list of alternate spellings usually allow any one of them to be substituted for the designated misspelling with a single keystroke. Most checkers also allow the correctly spelled words it has identified as possible misspellings to be entered to the dictionary via a single keystroke, after which they are not flagged.

At the present moment, spelling checkers are starting to catch up with word processing programs in terms of completeness. New versions typically have very much larger dictionaries. For instance, the new version of SpellStar (i.e., CorrectStar) has nearly 90,000 words in its dictionary,

making it more than four times as large as the (wholly inadequate) 20,000-word dictionary that came with the first version. With these updated versions, the user can at least be relatively sure the checker will catch almost all the outright spelling errors and many of the typos in a document, without bothering to identify many correctly spelled alternate forms of a base word (e.g., plural and past tense forms) as potential errors. In other words, these newer versions take the user through a text more quickly and identify a higher percentage of words that are actually misspelled. As grammar checkers too become more sophisticated, they will join with spelling checkers to provide a more thorough, although still incomplete, review of documents for grammar errors and stylistic weaknesses--thereby giving the writer a significant return for the time it takes to use these programs and make changes based on their findings.

For now, spelling checkers alone will be helpful to some students and rather useless, if not harmful, to others. (Having used spelling checkers for two years with my microcomputer/writing classes, I can say with some certainty that their utility will vary widely from student to student.) Those students who are not distracted from their text by the whole operation of checking its spelling (which often involves intriguing peripheral operations, like getting a total word count for the document) and who fully understand the necessary limitations of a spelling checker, will be able to locate and correct a number of spelling errors by using it. Often the designation of a word as a possible misspelling leads students to check it in their pocket dictionaries--and even if it is correctly spelled, they can benefit by having the correct spelling reinforced. For students who use them cautiously, spelling checker programs can improve their spelling, and even their word usage, to some significant extent.

Another, smaller benefit for some students in using a spelling checker is that it helps focus attention on their text in ways beyond the rightness or wrongness of its spelling. For example, VisiSpell puts the section of the text presently being spell-checked in a box on the screen with other information, including possible alternate spellings (or alternate word choices), outside the box. In scrutinizing the boxed text, some students will eventually notice things other than spelling that they want to change.

Those checkers that offer alternate spellings and alternate words do so strictly on a "character sequence" basis--that is, those words whose spelling is closest to the possibly misspelled word--and not on the (much more complex) basis of a word's meanings. Thus, spelling checkers are not too likely to improve students' vocabulary through the list of alternate words they offer--but they may help make a few of the more clever students aware of etymological similarities in words that are not normally linked because their meanings are so dissimilar.

The use of spelling checkers by faculty writers is likely to be less extensive than by student writers, since faculty users would have to spend some time building up the checker's dictionary to the point where it did not distract them by identifying a host of uncommon or specialized words as potential misspellings. On the other hand, faculty users would probably better understand the fact that spelling checkers are quite limited and that their employment is only the first of several steps needed to give a document an adequate proofreading. Those academic writers who have become very good

proofreaders through years of practice would probably be put off by the use of a spelling checker, and they would be right: they could do just as good a job themselves, and they could do several other proofreading jobs at the same time.

Like the calculator, the spelling checker's great advantage is the speed with which it performs operations that take human beings very much longer to do. For instance, it can check the spelling of each word in a 10,000-word document faster than a user with a printed dictionary could check the spelling of the fifteen words in it that might be in doubt. To the extent the user realizes and can utilize the kind of preliminary spelling check these programs provide, and go on from it to check for possible homonym errors, wrong words and incorrect endings, that user will find the spelling checker a real help.

The specific utility and sometimes perplexing operation of spelling checkers is exemplified by the results I had when using MultiMate to check a page of text. It identified two possible errors. The first was "summry," and it offered nine alternative selections, running from the very similar "summery" to the rather dissimilar "scummier" and including "samurai." The second choice, "summary," was the one I needed. The second possible error it identified was, for some reason, "this," and the first alternative it suggested was "this"! (The third, rather interestingly, was "Thais," which might make this a useful checker for those writing on French opera.) Redundancy, apparently, is yet another limitation of spelling checkers. Users who are able to put up with a fair amount of confusion, and some nonsense as well, will fare best with spelling checkers--since these do identify a certain number of "genuine" misspellings. Very few users will absolutely need to have a spelling checker, but many will find that having one can, in some limited but definite ways, help them avoid certain kinds of spelling errors.

Notes to Chapter 4

1. The most thorough review I have seen is in the April 1983 Personal Computing (110-129, 171). It analyzes seventy-four programs in relation to thirty-nine editing and forty formatting features.

1.1 The versions of the twenty word processing programs evaluated in Chapter 4 are listed below:

Personal QWERTY	DOS 2.0 version
Wordvision	DOS 2.0 version
Megawriter	version 3.0
Textra	version 2.12
PFS: Write	DOS 2.0 version
Select Write	version 1.0
Volkswriter Deluxe	release 2.0
Officewriter	release 2.0
Easywriter	version 1.1
Superwriter	version 1.03
Peachtext	version 2.10
Perfect Writer	version 2.0

Wordstar	release 3.30
Visiword	version 1.22
Spellbinder	version 5.30
Easywriter II	DOS 20 version
Wordperfect	revision 3.0
Microsoft Word	version 1.0
Powertext	version 2.0
Multimate	version 3.22

2. Both Townsend and Naiman do extensive reviews of a number of word processing programs for microcomputers. At the end of Townsend's reviews are checklists similar to those in the computer magazine reviews. Also available are introductions and guides to the more popular and complex programs: see, for example, Sommer or Naiman, "Introduction" under Works Cited.

3. Actually, I have tried to assess each program from four vantage points. Although she is speaking specifically about "Course-ware Selection," what Ann Lathrop says about testing holds equally true for word processing programs--namely, be yourself, be a "good" student, be a "bad" student, be a "negative" student.

4. Since there is a good deal of truth to the idea that the first program you use, particularly if it is an adequate one, conditions the way you think about word processing and establishes implicit norms for judging other programs, I should add that I worked exclusively with WordStar for the first year and a half of my word processing "career."

5. A few computerphiles do not, it should be noted, see EasyWriter as quite such a villain in the word processing melodrama. One of them even goes so far as to type it "a no-nonsense word-processing system designed, as the name implies, for the professional writer" (Schmeltz 150).

CHAPTER 5

WORD PROCESSING AND MICROCOMPUTERS: SOME SOCIAL, EDUCATIONAL, EPISTEMOLOGICAL, AND ONTOLOGICAL ISSUES

5.1 General Influences on Writing

Although the designers of word processing programs make a fair number of unwarranted and unjustified assumptions about the act of writing--some of which are scored in the previous chapter, they all share an assumption that is undoubtedly correct: namely, that you very likely will eventually need one word processing program or another (and so hopefully will choose theirs). Within a very few years many academics and many students will be using word processing for most, if not virtually all, their sustained writing tasks. In one way the matter is very simple: the time it takes to learn a word processing program and to become comfortable using a computer, as well as the continuing annoyances and inconveniences entailed by using the equipment (things like eyestrain and backache), will be far outweighed for most teachers, writers, and students by the many advantages of creating and reworking documents on a word processor. Even those holdouts who continue to produce documents by hand or by typewriter will probably find that their more important documents eventually get put on a word processor anyway, perhaps by the secretary who types the final draft (no secretary who learns word processing thoroughly will want to go back to a typewriter), perhaps when an article they have written is being prepared for journal publication.

In another way, though, the matter of word processing is very complex. Some see the word processor as merely another stage in the steady evolution of writing impedimenta: "From pen and quill through microcomputer, the tools of writing have constantly changed as the technology of the culture has changed" (Appleby 103)--but the whole history of the writing and production of texts does not bear out such a view. Rather it suggests that the major technological developments have created sharp breaks in the evolutionary cycle, after which the relationship of writer-text-reader reforms in an essentially new way. The development of the computer as a word processor may not bring about as dramatic a break as the invention of movable type in the fifteenth century, but its impact on writing and the production of texts could well be the most significant since that development.

Unlike even the most advanced typewriters, to which they bear a superficial resemblance, word processors produce text electronically, not materially. While most readers will continue to read computer-generated documents in the form of letters imprinted on a page, those "printouts" will only be a reflection of texts that are first and foremost electronic entities (created and preserved on a hard or "floppy" disk and only directly reachable via a computer). Some of the most significant and extensive ramifications of this break from the older way(s) of producing texts are discussed in subsequent sections of this chapter, but here we might first consider some of the simpler, typical ways in which word processing is likely to change the writing act and our perception of that act and its outcomes.

Somewhat paradoxically, word processors may reverse the trend away from written communication fostered by earlier electronic developments (most demonstrably the telephone) and mechanical inventions (automobiles which put us easily in direct contact with those nearby, airplanes with those far off). Because it is so much easier to "put it in writing" with a word processor than with other methods of written communication, and because it takes so little space to store ten of thousands of "pages" of electronic text, many persons, in that none-too-distant age where computers are available all around us, will inevitably be inclined to put more things into "writing." Offices will see more memos, formal (and informal) reports, even business letters. Personal users may find themselves "storing" more random thoughts (at least those of some scope) on the computer, where they can later be reworked and expanded. Academics will probably find themselves distributing more documents to students, since word processing makes it so much easier to create and, especially, to update things like reading lists, assignment directions, and reworked syllabi and class schedules. (In other words, they will find themselves doing in print, and so more definitively, what they often formerly did verbally and somewhat informally).

Already computer terminology and the kind of thinking it fosters have made a stylistic impact, almost wholly deleterious, on the language. A booster of word processing crows, "Throughput is greatly increased over the electronic typewriter" (Townsend 4), unmindful that the very sentence that trumpets this increased "throughput" is itself an ungainly, inaccurate sentence which should not have been "put through" in the first place. Too often--vide the remarks on PowerText--word processing programs encourage their users to treat the words they are writing as mere raw material for "processing" rather than as the very essence of, and the whole reason for, the communicative act in which they are engaged. Except for those in a secretarial position, "writing on a word processor," not "word processing" per se, should define their activity--a distinction that is sometimes hard to keep in mind when faced with the complex apparatus of a word processor.

Those who do spend a good deal of time at the word processor, whether out of need, fascination, or a combination of the two, are likely to experience some noticeable physical and mental discomforts, if not worse. Naiman's Buyer's Guide reports on a study that shows that using a word processor in the workplace markedly increased a number of unpleasant effects:

> VDT (video display terminal) operators at one or more work sites reported more (or more frequent occurrences) of the following:
>
> *Emotional problems
> *Gastrointestinal problems
> *Stiff arms and legs, swollen muscles, and pains in the wrists, back, neck and shoulders
> *Cramps in hands and fingers
> *Loss of strength in the arms and hands
> *Painful joints
> *Eye strain and other eye problems, like sore, itching, or burning eyes or loss of the ability to see colors
> *Anxiety, irritability, and "shaking inside"
> *Depression

*Anger
*Fatigue and periodic exhaustion
*Confusion
*Skin problems
*Headaches

> Workers who didn't operate VDTs reported many fewer problems. The only complaints they had more of than VDT operators were back pain, fever, arthritis, and rheumatism. (188)

Undoubtedly, the "mesmerizing" quality of the word processing screen, its fluid mobile resemblance to a television screen, makes it rather difficult for users to break away from it at regular intervals and so diminish or eliminate some of these problems. In addition, some operators report that using a word processor for several hours creates a strong sense of psychological disassociation from their surroundings.

At the same time, computers demand to be had and used. Advertising makes us so conscious that the product is there--the computer has replaced the automobile as the primary object every American family is shown to need--that it is difficult, in many "public" situations at least, to choose not to use a computer to write, or do other work, even when using it will be less efficient and more time-consuming. Indeed, computer advertising sometimes verges on scare tactics--like showing a toddler learning to peck out answers for an educational computer program, clearly implying that any children not raised from the cradle with a computer will be forever behind their peers educationally, technologically, even socially. Yet, to the extent we give credence to such a view, it represents the likely future of education in and through computers: many students will almost be forced to learn about computers to gain access to certain kinds of knowledge and to certain careers.

The sociolinguist Basil Bernstein points out how such socialization is a means of taming individual (and anarchic) impulses and assuring a certain uniformity of thought:

> Socialization sensitizes the child to various orderings of society as these are made substantive in the various roles he is expected to play. In a sense then socialization is a process for making people safe. The process acts selectively on the possibilities of man by creating through time a sense of the inevitability of a given social arrangement, and through limiting the areas of permitted change. The basic agencies of socialization in contemporary societies are the family, the peer group, school and work. It is through these agencies, and in particular through their relationship to each other, that the various orderings of society are made manifest. (162)

The basic agencies that Bernstein names--family, peer group, school, and work--are all likely to become "computer-related" encounters, for better or worse, over the next decades, at least for children in the middle and upper classes. Because word processing will be the most common use students and employees are likely to make of computers, it provides a demonstration, in

miniature, of how computers, particularly microcomputers, will influence and alter the quality of life and of learning.

One thing word processing will alter is the sense of privacy attendant on the writing act. As Zinsser notes, "the word processor makes writing a public act....We have become a society of television viewers, and the screen belongs to everyone" (44-45). Having one's words "displayed" the instant they are recorded--even if no one but the writer sees them most of the time--makes writing something more of a "broadcast" and something less a first recording of words to be reworked, rewritten, and shared only at the point the writer chooses. Similarly, the elaborate computer apparatus tends to signal the act of writing very clearly, just as jotting on the back of envelopes or in small notepads can obscure it, can make it part of a whole "flow" of activities. Thus, the microcomputer will help foster a sharper distinction between the act of writing and the many acts that are not writing, since even casual kinds of writing on a word processor involve one in a demonstrable series of machine procedures.

Provided one does not make the parallels too exact, nor insist that history will essentially repeat itself in the matter, the history of printing provides some clues as to what may happen over the next few decades because of the invention and dissemination of word processing. As Elizabeth Eisenstein points out in an important summary article on the history and influence of printing, "Increased output directed at relatively stable markets...created conditions that favored, first, new combinations of old ideas and, then, the creation of entirely new systems of thought" (8). As exemplified by much that has been written about word processing itself, the advent of "writing by computer" has already favored the kind of "new combination of old ideas" of which Eisenstein speaks. Often these combinations are new only in that they employ a discussion of computer technology as the vehicle for restating some common American cultural assumptions, like the vital importance of technology (and its moral potential), the significance of sheer productivity, and the concept of "time-saving" as an absolute value. Reading much of what has been written about word processing (and of other kinds of microcomputer programs designed for mass use) puts one in mind of what that very wise observer of the American scene, George Santayana, said about our national life: "In America there is a tacit optimistic assumption about existence, to the effect that the more existence, the better" (102). For uncritical analysts, word processing translates all too baldly into a system for writing more words more quickly and more thoughtlessly.

Whether the invention of the computer will lead to "entirely new systems of thought" is still an open question--although arguments for artificial intelligence as the virtual equivalent of human intelligence are becoming more convincing. Still, even the relatively straightforward technology of word processors contains potentials for radically altering the way we regard and create written things. For instance, if polymodality has been one of the recognized characteristics of advanced modern art in this century--typified by the all-inclusiveness of historical, narrative, and psychological scope in a work like Finnegans Wake, then the simple word processing device of allowing split screens, which puts two different texts on the screen and makes them one within that space, might make this "avant-garde" characteristic a rather common thing. Going a bit further, one might envision a text, or series of texts, which the reader--through the simple abilities of the computer to

alter, omit, and include so easily and quickly--would come to "make" for himself or herself, with the text of each reader differing to some extent from that of all other readers. Thus, the traditional notion of the writer as the sole transmitter and maker of a text and the reader as its mere receiver and decoder--a notion already under siege by "reader response" and allied critical approaches--would receive a further, perhaps fatal, blow. In light of such activity, the whole notion of what constitutes a text, particularly in terms of "finality," would have to be rethought (see 5.6).

When combined with other computer technology, word processing may well lead to new ways of getting at information and to new roles in the management and dissemination of information. As Eisenstein notes, with the invention of print technology, "The era of the glossator and commentator came to an end, and a new 'era of intense cross referencing between one book and another' began" (7). The potential computer access to millions of books, which could be had via a modem linking a microcomputer to a large main frame system, would lead readers, in a matter of moments, to many books they might not even have known existed. The period of "intense cross referencing" brought in by books will seem anachronistic in the new age in which readers can have books available on their computer screens in a matter of seconds and the sections of them they want printed out in a matter of moments.

On a more limited, but more evident basis, the coming of word processing has influenced (and will continue to influence) language--and language is the very stuff of social existence. One need not agree with the materialist vision of language presented by Marx and Engels to see how important a point they argue in stating:

> Language is as old as consciousness, language <u>is</u> practical consciousness that exists also for other men, and for that reason alone it really exists for me personally as well; language, like consciousness, only arises from the need, the necessity, of intercourse with other men. (51)

If language is the enabler of society, as Marx and Engels argue, it is also, therefore, the enabler of the social self--and so any invention that radically alters the way we go about creating language for others (and the self we create in doing so) is bound to influence language as a social act. If in no other ways, just the availability of screen text for public scrutiny and the endlessly potential changeability of the word processed text will significantly change the way we use language as "practical consciousness."

Something of the self-reflexive character of language is indicated by the fact that "to narrate" is traced back to the Indo-European root <u>gna</u>, meaning "to know." Many writers on computer subjects tell us that word processing is "simply" a means of making the production of written language "much easier," but about this facile judgment we should be suspicious: any method or device that so substantially changes the way we write must thereby change the language we create and the selves we create through language--and that change is no simple matter. The word processor, as it fits into a whole network of computer operations, will have a profound, if often disarmingly mundane, impact on the way people will be using, and will be used by, language in the coming decades.

5.2 The Future of Word Processing and Microcomputers in Education

Unless conditions change, poor students and students in the poorer schools (poorer both financially and academically) will not experience these changes as fully or as dramatically as their middle and upper class peers in the more selective American educational institutions. Whether or not "computer literacy" proves to be the boon many computer proponents profess it to be,[2] it will not fail to reach its potential through lack of opportunity in the better schools. Thus, computer literacy is certain to become a virtual requirement if one wants to continue in (or switch over to) these schools, and to secure a decent position after college graduation. Even now, anyone who aspires to a position in business must have some basic notion of the capacities of word processing to function efficiently in a business environment where the production of documents is a crucial part of almost any executive job. Soon, students at the better schools (not excluding the better primary schools) will be expected to do many of their writing assignments on a word processor, on equipment provided on site or on equipment students are required to have at home. Many of these institutions will provide computer resources that supplement the students' personal computers. (The degree of interface between institutional computer resources and personal computer resources has only begun to be explored.) Within a few years, writing on a word processor will become second nature to such students--and most of them will write more and write better, and so learn more, than students who do not have access, or have only very limited access, to computers. Thus, the already wide gulf that separates privileged and underprivileged students in our society could widen considerably, perhaps causing a certain segment of the population to lapse into a near muteness when it comes to written communication, the sheer "extra" labor of not writing on a word processor being the factor that provides the final discouragement. Since a fair amount of power is manifest in writing--the law being only the most overt example, these citizens could become virtually powerless through this muteness.

It is difficult to pinpoint, or even categorize, the precise motivating forces in this scenario of a society increasingly divided, to borrow Henry George's distinction, into the "House of Have" and the "House of Want" in terms of computer accessibility and knowledgeability. Rather, we have in the process an instance of the extremely complex and mutually reinforcing ways power comes to be exercised in industrial and post-industrial society. The situation very much resembles the general outline of power given by the French historian of ideas, Michel Foucault:

> ...power must be understood in the first instance as a multiplicity of force relations immanent in the sphere in which they operate and which constitute their own organization; as the process which, through ceaseless struggles and confrontations, transforms, strengthens, or reverses them; as the support which these force relations find in one another, thus forming a chain or a system, or on the contrary, the disjunctions and contradictions which isolate them from one another; and lastly, as the strategies in which they take effect, whose general design or institutional crystallization is embodied in the state apparatus, in the formulation of the law, in various social hegemonies. (92-94)

The complexity of such "force relations" is evident when one considers the question of which students in this country are now being given access to computers and training in their use--and who is paying for the equipment and instruction. In one way, the producers of computer hardware and software cannot be faulted, since they have reduced prices markedly over the past few years in seeking a mass market. (But they can be faulted for still selling their products at a very large markup).

Many have instituted a policy of even further substantial discounts for educational users. Yet, things being what they are in a capitalistic society, it is precisely the wealthier educational institutions that can take, and have taken, best advantage of the situation, since they can usually find the capital to finance large-scale hardware and software purchases. Poorer institutions rarely have the fluid capital available for the purchase of many computers, and often must rely on uncertain grant monies or cut other programs to make computer purchases--and when they do buy equipment, they do not buy enough to serve all their students.

When it comes to requiring students to buy their own microcomputer through some special arrangement the school has made with a computer company, the situation is much the same. Consider this: There are presently between twenty and thirty colleges and universities in the United States where a year of higher education, including room and board, costs over $14,000, with M.I.T. currently the highest, at over $16,000. Needless to say, requiring students at such institutions to spend the $2,000 or so it would take to give them a personal computer for use over their four years in college is hardly adding a great expense to the overall bill. Moreover, such institutions typically have extensive computer resources besides, resources that students can access and use via their personal computers, thus making their own machine an even more valuable and "cost-effective" purchase.

At the other end of the educational spectrum are institutions whose students must strain just to meet much, much lower tuition costs--or perhaps who come from families whose income is so low that most or all of their tuition is covered by some sort of aid from the state. For a majority of these students, $2,000 for a microcomputer would represent a very considerable, probably prohibitive, additional cost, one that could effectively double or triple the yearly cost of their higher education. Therefore, many institutions that have predominantly lower middle-class and poor students could hardly mandate the purchase of microcomputers by students without risking mandating themselves out of existence. And while many such institutions--my own included--have made notable efforts toward institutional purchase of computers and the integration of work on microcomputers into the curriculum, such efforts will remain piecemeal until these institutions receive an inflow of monies that would allow them to start planning for the purchase of hundreds, even thousands, of the machines they are now, if they are lucky, buying a dozen or two of at a time. (Another difficulty for institutions squeezed financially is that they also need to finance new space for computers.) Even then, almost all these institutions would lack the sufficient large-scale computer resources which would make the students' microcomputers more flexible in operation and more extensive in coverage, and the more valuable thereby. In short, the coming of the computer, particularly the microcomputer, to higher education is very likely, unless things change radically, to exacerbate the split between wealthier and poorer institutions,

since it is a development that demands, as things like innovative pedagogical systems often do not, a very substantial monetary investment at the outset.

Although secondary schools, then probably primary schools, will eventually become the places where students learn word processing (or at least learn it systematically), most of the present generation of college students will have to learn it as undergraduates or risk going without what will soon become virtually a basic skill. Those who are teaching word processing, and particularly those who are teaching it in a content area course, will need to create a pedagogy that complements the instructional documentation that comes with the program being used. The better word processing manuals are clear and explicit about how to perform any number of their program's operations, but none of them gives a good, practical sense of how to use word processing in a larger context--i.e., as part of a whole writing approach and system. Based on, but not restricted to, their own experience, teachers will need to analyze carefully just how, in their subject-area writing, a word processor can augment the production of clear, precise, intelligent documents. And then they will have to translate this understanding into explicit, pedagogically sound instructions, exercises, and assignments for students, always allowing for the considerable variation in writing styles (including the haphazard) that one will find in any group.

This knowledge of how to use word processing--as opposed to knowing merely how to operate a word processing program--may well soon form a rough equivalent to the kind of "meta-language" Bernstein describes as one of the key sociolinguistic factors that separate the powerful from the powerless:

> Historically and now, only a tiny percentage of the population has been socialized into knowledge at the level of the meta-languages of control and innovation, whereas the mass of the population has been socialized into knowledge at the level of context-tied operations.
>
> A tiny percentage of the population has been given access to the principles of intellectual change whereas the rest have been denied such access. This suggests that we might be able to distinguish between two orders of meaning. One we could call universalistic and the other particularistic. Universalistic meanings are those in which principles and operations are made linguistically explicit, whereas particularistic orders of meaning are meanings in which principles and operation are relatively linguistically implicit. If orders of meaning are universalistic, then the meanings are less tied to a given context. (163-64)

It is just this <u>universalistic</u> understanding of how to use word processing, regardless (or even in spite of) the <u>particular</u> program or system being used, that constitutes the most difficult and the most necessary thing teachers, especially at the less privileged schools, need to impart to their students. The kind of "control" and "innovation" in writing that word processing can abet is something no student can learn from a program manual.

As computer learning and writing on computers come to take on more and more educational importance, those who teach underprivileged students are likely to find themselves on the horns of a dilemma, caught between advocating equal computer accessibility for their students and decrying the often inadequate and inappropriate software available for use on the equipment. Helen Schwartz raises, if in rather too reductive a way, a crucial point concerning the use of computer programs in education with this set of questions:

> Does the program respect the human rights of the user?
>
> --Can the student sign on and off to fit his or her interest and time?
>
> --Is the program "polite" in the tone of its responses? ("Monsters" 150)

She might well have added a third, and more significant, question: Is the software being used relatively free of intellectual coercion, thereby allowing the user to develop freely along intellectual lines? The responsive power of the computer is predicated on its ability to enforce certain decisions a priori, and so it is important to assess software in terms of the limits and flexibility of the assumptions upon which its operations are based. Undoubtedly, computers will make learning more impersonal and less context-specific, but they need not make it culturally monolithic.

The use of word processing provides one compromise between introducing students to and making important use of computer resources and still leaving sufficient space for teachers to tailor instruction and assignments to the specific needs and situations of their students. Although they act on a number of assumptions about the cognitive and composing processes, word processing programs are not directly instructive. To make them effectually interactive, instructors must create a pedagogical framework for their use, employing their particular capacities (like the ease and speed of revision) in a way that makes them powerful adjuncts to the learning and writing processes. In addition, instructors can make word processing programs active "instructors" through such devices as adding "help files" (on specific writing programs, with lists of suggestions) to students' disks, where they can readily be called onto the screen at the students' discretion.

Because they are such a prime market item, personal computers (and to a lesser extent, word processing programs for them) are surrounded by a whole sales rhetoric--Reinecke is more right than not when he observes that for sales purposes "microcomputers are represented as moral improvers" (77). This rhetoric must be penetrated if microcomputers are to be used effectively in education and not merely scattered widely through educational institutions, with students left to learn and use them as they will. Moreover, what computers can and cannot do must be assessed in terms of the varying groups of students (rich, poor, with sound basic educations or not) who will be using them. At the present time, it is probably also necessary to devote some class time to explicit analysis and discussion of this whole rhetoric surrounding the computer, so students will have a more realistic sense of what the equipment can do and how it can most effectively be used, things that the sales rhetoric glosses over or ignores.

My own pedagogical experience confirms the rightness of James Kinneavy's judgment that:

> Minority students see, sometimes more vividly than other students, the massive effects of commercial, racial, and political rhetoric on their lives. Advertisements in grocery stores, which we often interpret as merely informative, strike them as coercive and unfair, for example. ("Restoring" 26)

There is, when we look at it, a decidedly unequalitarian bent to the sales rhetoric of computers, a bent that is sometimes ignored in the high flush of enthusiasm for the machines and in the concern to make educational institutions computer-ready. Many minority students, as well as those from poorer families, do not need much prompting to realize how, to a large extent, the computer revolution is not shown as touching on or mattering in their lives. For instance, the computer users pictured in advertisements, whatever their color, are unfailingly middle (or upper) class and often, if the setting is an office, they are shown as ingenious and powerful types (most of them men, save for the women in secretarial postures). In fact, the computer is depicted in many ways as the emblem of a modern, upwardly mobile life--a depiction minority and poor students may come to realize, probably for their own betterment, is one of the covert, but troublesome, attractions to them of a computer-related career.

For the moment, then, a teacher who brings the microcomputer into the classroom must do three things that will help his or her students to learn about computers, about their present status in society, and about the place they will assume in the coming society:

1) Make students aware of the powers and limitations of the kind(s) of programs they are using--and of microcomputers and other computers in general. Students need to be taught to use computers and not just how to operate them.

2) Make students aware of the duplicities and evasions--as well as of the degree of truth--in the popular rhetoric of computers. It is particularly important to demystify program operations, to explain in as simple terms as possible why a program can perform functions that can seem, and are often presented as, "magical."

3) Make students aware that their (hopefully) clear-eyed view of the computer will not usually be the norm, for the near future at least, in many of the business and other situations they will find themselves in after graduation. They may have to blend their own realistic vision of computers with an unrealistically enthusiastic view that holds sway and determines the shape of computer-related work where they are employed.

Elizabeth Eisenstein observes that one of the most profound effects of print technology, which so dramatically increased the number of copies of a text that could be in circulation, was a thoroughgoing restructuring of the educational system--and a consequent reconceptualization of the early stages of life:

> Possibly no social revolution in European history is as fundamental as that which saw book-learning (previously assigned to old men and monks) gradually become the focus of daily life during childhood, adolescence, and early manhood....A new "concept of childhood" indeed might owe much to the widened gap between literate and oral cultures....The appearance of a stricter domestic discipline, together with new forms of child-rearing, schooling, and worship, was probably linked to the inculcation of book-reading habits. (41)

One suspects, again, that in the socio-educational area, the computer revolution will not be as total as the print revolution--but that it will be very significant nonetheless. The video display screen will combine with (and occasionally vie with) the instructor as the primary vehicle of instruction. As a result, learning will become a more self-paced activity, with the student choosing the speed and direction, and with the instructor monitoring student learning and contributing additional instruction to it (often on an individual or small group basis). Education will also become more piecemeal, if somewhat more efficient thereby, as students and teachers select the various combinations of materials each student will cover, after which computers will provide students with just the material requested. (Time undoubtedly will be saved, but students will miss some of the information they pick up casually as they browse through books and articles looking for the specific information they need at the moment.) Libraries will gradually decline in importance as many of their users acquire personal computers that can be hooked into large computer banks containing millions of texts. Eventually, libraries will possibly reach a still point, as the repositories of some electronic texts and computer equipment, and as the storehouses of those texts readers will still need, or prefer, to consult in printed form.

One of the reasons that the computer revolution will not be as total as the print revolution is that electronic texts will not, on the public level, displace printed texts as completely as printed texts replaced hand-lettered ones. The space limitation of the video display screen, even if the amount of text that could be shown were increased three or four times (a direction in which the technology seems to be heading), is simply too inhibiting when it comes to reading and working with documents of any length and complexity. Therefore, most users will want to deal with such documents as printouts, i.e., a form that resembles a cross between typed and printed text. Moreover, documents intended for a large public will still be printed in much the same way they are now (although, to be sure, word processing and other computer technologies will figure centrally in the process).

Despite many similarities in appearance between printed and computer-generated texts, the impress of computer technology on print technology will be great. Word processing technology, which is at once among the simplest computer technologies in overall operation and among the most widely used, is subtly changing the way we think about "writing" and the "documents" and "texts" produced by that activity. Present-day teachers, who have been educated and have worked under the older print technology, can play a key role in educating the present generation of students about the effects and implications of the ongoing, irregularly implemented transition to electronic texts. As a start in that direction, the remainder of this chapter

deals with speculations on four ways the coming of electronic texts, in general, and the employment of word processors, in particular, may influence and restructure our notions of the whole process of written communication and the kinds of products that result from it.

5.3 Writing as an Electronic Act

Word processing makes composing, at some point, an irreducibly electronic act. A piece of writing is entered as a series of electronic signals and only exists in something approximating conventional printed form after further electronic acts, usually, the command to "save" the text by recording it on an electronically scannable disk and the command to print that text from the disk. Both these operations involve the kind of "machine-to-machine" transfer of signals that the semiotician, Umberto Eco, characterizes as outside the realm of human communication, therefore having "no power to signify" (8). Thus, word processing technology involves us, inescapably, in a series of information transfers from machine to machine (or from one component of a machine to another). This factor creates not simply, if significantly, the possibility of technical malfunction, but also delimits the "written" text as a thing wholly and completely "electronic" in its essence. The "human" element typically enters the process only at the point of the text's origin (as the user gives a series of electronic signals to enter the requisite characters) and at the point of the text's destination (when the creator or someone else reads it in printed form). In between, the text is a complex electrical entity. Furthermore, the text is never available for reading without some form of electronic technology to reproduce it--either the video display terminal or the printer. In an important way, the "original" of a document is, by human standards, inaccessible, since it consists of ten of thousands of "unreadable" electronically made markings on a chemically treated piece of plastic or aluminum. It is, in a very real sense, only the machines that deal with original texts, the users only and always seeing a reflection of the originals. The situation of reading the computer printout of a document, then, is rather like the situation of confronting a "sculpture" (say, Rodin's of Balzac) made from a mold: the viewer sees only a copy, of which there might be one or a thousand, but seeing the "original" (the mold or the disk) would be meaningless in terms of any human communicative process. In Eco's terms, the original has "no power to signify."

The philosophical implications of this new technology, with its electronic originals, are significant, if generally ignored. For instance, one word processing enthusiast notes how, as a matter of great convenience, "Book manuscripts that I prepare for a publisher can be sent to a typesetting machine by telephone" (Townsend 8). He might have gone on to note, paradoxically, that a manuscript thus transmitted in such a machine-to-machine transfer (through yet another machine, the telephone) involves the transmission of a "more original" copy of the manuscript than sending the printout to the publisher would have. This point is evident when we realize that after the electronic transfer the typesetting machine would have in it the whole manuscript, capable of being printed immediately (although hundreds of typesetting directions would still be needed to prepare the text for printing in the desired format). The publisher who received the printout would only have a reflection of the original, and would thus have to create a whole new "original" on the typesetter.

With word processing, then, the printed form of a text is only a secondary, not a primary, form of its existence. Every keystroke that makes up a document exists only electronically on the primary level, temporarily as entered to the computer's active memory and permanently when recorded for storage on a disk. In this respect, even the electric typewriter is closer to primitive forms of calligraphy in the way it produces documents than it is to the word processor. With electric typewriters, electricity only functions as an aid in producing a text materially and directly. If the power goes out on an electric typewriter, one has a physical copy of every character typed up to that point; if power goes out on a word processor, one has only that portion of the text that has been specifically "saved," and has that only as, in terms of a printed text, a potentiality. It is even possible that because of a disk or system error the text might exist as pure, unretrievable electronic potentiality, a recording of words unable to be called up to the screen or marked down on paper by the printer. Anyone who has worked with word processing for just a few hours will come, if only implicitly, to this important recognition: in return for physical ease of construction and reconstruction, the word processor enforces a certain insubstantiality on the text, since that text is first and foremost an electronic entity.

There is a scholastic formula that holds nihil in intellectu quod non prius in sensu. ("There is nothing in the intellect that was not previously in the senses.") With the advent of electricity the self-evident quality of such a epistemological truism became debatable. To some extent computer designers have recognized the psychological need of computer users for sensory experience that reifies their intellectual-motor activity. Thus, almost all microcomputers give off a low click as keys are pressed, althougn the keyboard could just as easily be silent. (Some programs allow this sound to be turned off, or increased in volume). The need for this artificial auditory feedback is probably twofold: 1) it makes the operation of the computer keyboard resemble that of the typewriter keyboard, thus reinforcing their similarity, and 2) it provides a sensory correlative for the keystrokes that are being recorded electronically by the computer. For touch typists at least, the continuous sound of the keys being struck is more a certification that the document is accreting than the visual appearance of the characters on the screen, since they have been trained not to look at the text as they produce it. On the large question of just how a word processed text exists, however, there is little that can be done to increase the sensory substantiality of the text. Many users of word processing eventually come, if only half-knowingly, to the disquieting conclusion that what they write on a word processor is of an intrinsically transitory nature. Zinsser hints at a certain ontological dichotomy word processing enforces between the creation and existence of a text: "With my new word processor I would type my words and see them materialize on a screen. At that moment the words would, I suppose, exist. But would they really exist? Not in any sense that I had ever thought of words before. They would be mere shadows of light" (21).[3] Since we are used to thinking of the words we handwrite or typewrite as having an absolute existence in and for their text (they are discernible ink marks recorded one at a time on a page), it may be difficult for us to adjust fully to the indeterminate status, as "mere shadows of light," that word processing imposes on our words.

Even in advocating its major advantages, some writers on word processing convey the sense of unreality attendant on this electronic type of writing:

e.g., "A great power of writing with a word-processing program--perhaps the single greatest power--is the ephemeral nature of the text. It's not really there. It's a fantasy. So you can sort of dream it and redream it as you please" (Gonzales 61). However much we tend to think of texts as compounds of ideas and abstractions, we also, in another sense, tend to think of them in corporeal form, as a series of ink impressions on paper. Word processing does eventually put most texts into this form, but only as a corollary to its electronic recording process--and it is just the vague status of word processed texts at the moment of their creation (and recording) that makes them problematic ontologically.

Pondering the scholastic formula quoted above, Leibniz was prompted to amend it by adding nisi intellectus ipse. ("There is nothing in the intellect that was not previously in the senses except the intellect itself.") With the invention of computers, the formula might have to be amended further (I do not volunteer the Latin phrasing) to suggest how computer programs can function as a kind of secondary "intellect," less important and separate from, the primary intellect of the computer use. Even the relatively straightforward operation of word processing programs demonstrates this "dual intellect" condition. For a simple example, consider the probable evolution and use of those "dot commands" employed by a number of word processing programs. First, a program designer "told" the program something that every writer realizes, if only subconsciously: that a period can never appear as the first character in a line. Then, the program was instructed to treat each such appearance not as text but as the signal to enact a formatting comment, the exact nature of which was to be specified by the characters following the initial period. Users of such programs--in order to perceive "." correctly--had to add a new kind of sensory recognition. Besides contextual location (most commonly at the end of a sentence), they now had to consider its location within the electronic text as that text appears on the screen; namely, that a period in the first line space has a wholly other, nontextual meaning from its appearance in any other space. When the user's intellect and the computer's "intellect" mesh on this matter there is no problem, but when they do not, there can be a clash: e.g., the user who accidentally types a period in the first line space could end up with part of a text formatted in an undesired and perplexing way. The "intellect" of the program--which always recognizes a dot command when it sees one, but which is incapable of judging its feasibility or considering the possibility of its accidental creation--does, in fact, take initial precedence, with the user's intellect only able to take final precedence through a series of adjustive maneuvers.

Consider, in a somewhat more complex way, how the user's intellect and the program's "combine" to handle certain things on a general and specific level. For instance, the user can instruct the program to avoid, in dividing a file into pages for printing, "widows" and "orphans"--i.e., never to place the first line of a paragraph at the bottom of a page or the last line at the top. The user gives the general command to proceed thusly, but it is only the computer which considers and acts to avoid each possible instance. The user knows what is wanted as a general rule, but it is only the program that "knows" the instances where the rule is applied.

Of course, these examples from word processing operations are only of the simplest kind when compared with the most advanced and complex forms of "intellectual" work done by artificial intelligence. Still, they do point to

the degree to which the computer's memory and capacity to act is separate from the intellect of the user. New users of word processing often get the impression that the computer is doing very many things to their text (automatically moving to new lines, justifying margins, setting the number of lines per page, etc.) of which they are only half aware. They are right: whatever thought users must apply to the content of a document, the word processing program has already pre-thought and predetermined a number of things about its arrangement. Under such circumstances it is difficult to maintain the belief that a document's creation is wholly and simply the product of its writer's intellect. Although the exact sensory status of electricity may be, as I understand it, a debatable thing--consider, for one thing, the function of electrical impulses within the brain--electricity is undoubtedly a sensory phenomenon and so what a computer "knows" it can be said to know based on the priority of its sense experience. The real question, then, about computer "intelligence" is not whether computers "know" (in any absolute sense of the term) but rather if what they know can be communicated in ways that approximate human communication--but to answer that question we would ultimately need to go too far afield from our topic of word processing.

5.4. Word Processing and Pseudo-Consciousness

We might recur to a point in the history of print technology to provide a contrastive instance with current (and probably future) developments in computer technology. If the mid-eighteenth-century expansion of the textile industry provided sufficient rags to start a great growth in publishing activities, it was early nineteenth-century developments in print technology itself that accelerated that growth to the point where the nineteenth century became the first age of mass circulation of printed documents:

> The printing press sought out and found a form of motive power different from muscle power. Between 1803 and 1814 Koenig built three types of machine which were the prototypes of modern printing plant: the power-driven platen press, the stop-cylinder press and the two-revolution press. As early as 1791 an Englishman, Nicolson, had worked out the principle of the cylinder steam press with self-inking rollers. Inventions like these of course greatly accelerated the output of books and prepared the ground for the newspaper, the most recent newcomer yet to the world of print. The newspaper! It characterizes the grip printing had on readers in the late 19th and early 20th centuries. All such inventions were the result of great social transformations, but in turn gave further impetus to their development. (Febrve and Martin 12)

Yet, as this summary indicates, the much improved machinery still replicated "muscle-power" in the method of its operations, however much more durable and faster it might be. With computer technology, however, the paradigm of imitating and perfecting muscle (even brain) power disappears. When certain operations, like checking a 5000-word text against a dictionary of 90,000 words, are done thousands of times faster than any human mind could perform the operation, and when such operations are, for the most part, done invisibly (i.e., not reflected on the computer screen), the question of replication of

human effort is essentially obviated. Furthermore, things are done by computers in ways that have no practical parallel in human activity--e.g., the thirty or so seconds it takes a word processing program to record permanently an edited version of a twenty-page text containing several hundred changes, can in no real way be compared with the several hours it would take even a skilled typist to produce a "clean" version of the edited text. Moreover, the pre-edited version has itself (under normal circumstances) been omitted from electronic consideration--it no longer "exists."

Perhaps the most obvious manifestation of the difference between writing on a word processor and writing any other way is that the newer method involves a sort of dialogue with the equipment. Many new users of word processors may find, as did Zinsser, that they "hadn't realized that the machine would have an identity of its own, that it would be able to talk back" (16-17). Some users may judge, as did one instructor who has worked with CAI in composition classes, that this dialogue has a generally positive impact, particularly pedagogically:

> My experience is that most students treat the computer as a friend. They are comfortable talking to it and are awed by its responses. They may argue with a teacher that "everyone knows" what a fuzzy word means, but they do not argue with the computer, which responds to a short answer with "Tell me more." (Arms 356)

As noted in the opening chapter, there is a prevalent tendency in academia to "humanize" the computer; often its ability to talk back is taken as the emblem of its potential to counterbalance the sterile, "mechanical" impression created by its extensive, often intimidating equipment. One instructor who has used the computer with students stresses how it is "consistently warm. Its comments are pleasant and, more important, it never loses patience" (Kreiter-Kurylo 180)--all of which makes it sound very "understanding" in the larger sense of the term. The writer of one of the sounder and more realistic appraisals of the employment of computer programs in higher education indicates just what his critical stance will be in the opening words of his essay: "Like many humanists..." (Borque 67). An important designer of CAI indicates a similar perspective in speaking of how programs must be developed "in accord with humanistic teaching methods" (Helen Schwartz, "Monsters" 141). All too frequently--and all too easily it seems to me--the still much-debated, oft-rejected, and variously understood concept of "humanism" is invoked as the only thing that needs be applied to or blended with the technology of the computer to make it a near-perfect teaching instrument.

What is too often ignored, however, is that at least the simpler kinds of programs, like word processing ones, are specifically not modelled on human consciousness, but instead merely take on that guise at certain points in their operation. What the user of a word processing program must confront and deal with is a pseudo-consciousness, which presents the results of a long chain of split-second binary operations in the syntactic form of a reaction to the user's keyboard actions. The following false analogy typifies attempts by the wholehearted proponents of computers to force a resemblance between basic computer operations and those of human intellect:

> If you think of your mind as a sort-of-computer, when you write even a simple sentence you have, in essence, processed words. The procedure goes something like this:
>
> 1. From your vocabulary, stored in your memory, you choose words that generally fit the subject you wish to write about.
>
> 2. You select the best of these words.
>
> 3. You select words to connect the best words together.
>
> 4. You lay them out with a beginning, a middle and an end based on certain rules you have learned, again calling on your memory.
>
> You have successfully processed words into a sentence. You might then change and alter the words, the arrangement of words, add words, delete words--all of which would be a continuation of the word processing process. (McWiliams 22-23)

The demonstrable fact is that the human mind does not work in such a straightforward "search and select" pattern in constructing statements--i.e., the speaker-writer does not run through the thousands of words in his or her vocabulary to choose each one for each slot. Rather something that is both vaguer and more complex occurs: the ideas the speaker-writer wants to express, which already have a nascent existence as words in the mind, suggest, within syntactical parameters, the exact words that might be used.

Similarly, when the computer communicates with the reader it does not search its total memory of words to form a combination, but rather, based on a precise computation of circumstances, puts forth one or another of a relatively limited number of prewritten messages. Therefore, using a word processing program is only an interaction with a pseudo-consciousness. For example, many programs will ask, as a safeguard, "Are You Sure (Y/N?)" before enacting any large-scale deletion (like that of a whole file)--and they will always ask that question, even after the user has performed thousands of such operations (it has no way of calculating the user's experience) or even when the user seems to misunderstand the question (e.g., if he or she does not press Y or N but keeps repeating the command to save). What appears as something of a communication--the final confirmation request--is really just the last step in a multi-step deletion process.

Although they are not highly interactive, word processing programs do contain a number of such messages, messages to which users must react if they are to make the program work. The frequent injunction to "Wait" while the program is conducting some operation must be heeded, lest an entire file (or entire disk) be damaged and text made irretrievable thereby. Users are constantly offered choices in the guise of communication: "To see further pages of the J Menu press..."). More than occasionally these messages are inexplicable, particularly to new users, but the machine cannot be asked, as a human communicator could, to explain itself. (The most common resource in

such situations is to revert to the older technology of print and consult the list of error messages or other information in the manual.)

Still, it is a bit difficult to remember while working with a word processor that one is not dealing with a reactive consciousness, but only with the mere expression of the results of a vastly complex set of binary functions. Yet, to reach a real consciousness--i.e., at least one reader to receive and interpret your message in printed form--you must respond to this pseudo-consciousness. All of this interaction with a pseudo-consciousness is bound, I think, to interfere somewhat with the image of one's posited, potentially real readers. The user must write, in one sense, for the program's approval, so as to reach beyond it to other "readers."

Because their responses are programmatic, schematic, and limited (especially compared with those of an average human auditor), word processors can safely be said to represent only a pseudo-consciousness--although when we move on to the more complex kinds of CAI, which allow "branching" (i.e., a variety of possible responses to each communication from the user), we can be less sure that we are dealing with a pseudo-consciousness. With such programs, however, we are still getting a considerably narrower range of responses than we could expect from a person. How many, if any, of these programs could suddenly decide to change the topic? And if they could, what would they change the topic to? Still, there is, theoretically at least, some point at which computer responses could simulate (or even bypass) the number and variety of human responses to a statement or event. As A. J. Ayer observes, if the dividing line between pseudo-consciousness and real consciousness (the basis of "personhood") is the criteria of "experiences," these are things computers may eventually be able to have:

> Suppose now, what is not so far the case, that machines were to satisfy the criteria on the basis of which we ascribe experiences to one another. I regard this supposition as improbable but not as logically impossible. If it became true, then our only reason for denying them the full possession of consciousness would be on the score of their origin or their physical composition. I do not see why they should be disqualified on either of those grounds. I think that we should treat them as persons with a peculiar constitution. I am suggesting not only that this would actually be our reaction but also, more hesitantly, that it would be justified. (16)

We must not, however, confuse the higher (and still largely potential) developments of computer technology with its present status in lower forms like word processing. Virtually every "communication" a word processing program puts in the form of an interrogative or imperative sentence could easily be turned into a (more neutral) declarative statement with no substantive change in meaning. (The fact that they are more effective as questions and directions does not change the matter.) This proclivity of word processing programs for the interrogative and imperative is, we must realize when using them, just one of these "constraints built into the context...in which we find ourselves operating," to cite Stanley Fish's characterization of what must be understood when interpreting any message (268). Unfortunately, for the moment, the constant hard sell of computer advertising makes it

difficult for us, as teachers of students using computers and as individual computer users ourselves, to treat word processing not as a form of communication, as "user friendly," but rather as a new, in many ways radically different, and impersonal electronic means of producing texts. Word processors are revolutionary not because of their superficial resemblance to persons who question and guide us through a series of operations, but because they produce, handle, and restructure documents in a way and with a speed that surpasses, sometimes a thousandfold, the speed with which the operation can be accomplished manually or by other mechanical means.

5.5. Word Processing and the Ontology of Writing

Steven Marcus points to the rather indeterminate status writing assumes when it is done on a word processor: "There are indications that print which appears on television screens is neither print, per se, nor television. Rather, it is print-on-television: a new medium with its own characteristic messages" (162). Thus, the moment-by-moment creation of a text at a word processor is also, in some degree, a "broadcast" of that text. (In a classroom where terminals are linked so students can see each other's texts this effect is heightened.) Conversely, the malleability of the word processed text makes its transmission, even in the most finished-looking form, something less than definitive. The word processed text is both more readily and continuously available for scrutiny by others and less stable and complete than a conventional printed text.

To the extent that one composes directly on a word processor, one subverts the traditional cultural hierarchy of writing. Writing in the most basic sense means "handwriting," with the signature ("to write one's name") its most elemental form. Above handwriting is the mechanical writing of single copies of texts on typewriters--typewriters can be the point of textual copying (from holographs) or the second-level point of origination of texts. In the latter case the writer is already beginning the subversion of the writing hierarchy. At the top stand the various forms of published writing, ranging from the relatively impermanent newspaper to the relatively permanent hardcover book. Of course, the development of fast, efficient photocopying has already destabilized this hierarchy, by allowing the typed text (even occasionally the holograph) to be the version that is replicated and disseminated (i.e., "published" in a limited sense). The word processor, however, is by itself a unit for recording, revising, and "publishing." To record a text on a word processor is also to make that text instantly "publishable" via printout, and publishable in as many copies as ordered (though printing many copies of even a medium-length piece would be a very impractical use of the computer's resources). Moreover, the normal method of word processor operation essentially does away with the idea (and existence) of successive "drafts"--i.e., as one enters and makes permanent corrections and revisions, the update version of the text is, electronically and hence reproducibly, the only version of the text that exists (printouts of early drafts notwithstanding). The word processor, then, turns the traditional hierarchy of writing--which in full-fledged terms moves toward clearer and clearer versions of a text, each a material precursor of the material version that will be finalized in print--into a simultaneity. What is recorded can be printed at any moment; what is printed at any moment is never permanently fixed.

One practical result of this destabilization is likely to be some destabilization of the publishing industry. The development of print technology soon saw the concentration of publishing in the large commercial centers, with the further developments in the technology and the increase in the reading public tending gradually to squeeze out smaller publishing concerns. In our period, many of the larger publishing concerns have combined with or been absorbed by their competitors. The number of newspapers has dwindled markedly, with many of those surviving now being part of large publishing chains. In the last few decades, some of the large publishing firms have, in their turn, been absorbed by megacorporations many times larger still. The result has been virtual control over the published documents that reach a large readership by a fairly small number of companies.

While the combination of word processing, allied computer technology, and inexpensive photocopying will not necessarily do anything to break down the vague form of censorship that results from this monopolistic publishing situation--as Foucault would summarize the situation, "power is exercised from innumerable points, in the interplay of nonegalitarian and mobile relations" (94)--it will certainly tend to blur the distinction between a published and non-published document. As word processors become more and more sophisticated, they will make it easier and easier for users to produce documents that virtually resemble, and so share in the traditional authority accorded, the printed word. Since such documents can so easily be revised, with each new version replacing the old <u>in toto</u>, a certain revisionist tendency will inevitably emerge: at least electronically, the present version of a document will be the only one "in existence." On a more practical level, the finished look of word processed documents may cause some psychological readjustments when we need to consider them as still only very rough drafts. Consider the possible conceptual difficulties in trying to give a critical reading to the first draft of any essay intended (many drafts later) for publication, when its very "roughness" is masked by the finished look of the piece--visually it already very closely resembles what it would look like when (and if) printed.

Word processing will also, to some extent, decentralize printing. It will allow, and so encourage, small-scale "printing" on a local, or even an individual, level. A new class of document--something between a typewritten document and a published one--will arise. To a limited extent, the possessors of word processing equipment will become their own "publishers." Documents needed in several hundred or several thousand copies--just the numbers that normally make professional printing economically unfeasible--will be able to be printed "at home." The appearance and visual variety of these documents, if they are done with a sophisticated word processor, will make them look more like published texts than typewritten ones--and their content will gain a greater credibility thereby.

Whether the spread of word processing and computer technology, which will undoubtedly bring about some conflation of the ideas "to write" and "to print," will lead to "entirely new systems of thought," to recall Eisenstein's analysis of the long-range effect of print technology, is as yet unclear. What is clear is that the impact of the computer, particularly the personal computer, on the way we write and how we think about writing will be considerable. Right now we seem to be in some confusion, because word processing resembles, at the point of creation, both the typewriter and

television, and, at the point of physical copy, the printing press. For the moment, then, writing on and reading the output from a word processor often creates that effect Eco characterizes as "noise," namely, interferences with the older systems and forms of communication, the mixture creating something of a "new culture" in which communicative signals take on ambiguities and the outcomes of the communicative process are, as yet, uncertain and variable (141-142).

The enormous quantitative difference in cost/time between redoing a conventionally printed text and a word processed one translates into a qualitative difference. Thus, one of the unspoken postulates of a word processed text is its literal conditionality: it presents itself as something always capable of being rewritten and reformed, since the labor involved in doing so is minimal and the time needed to produce a whole new version is very short. Moreover, the ability of the word processor to generate finished-looking text does not, in most cases, certify the same thing about that text that such an appearance does with a conventionally published text: namely, that a fair number of people with some editorial expertise have looked at and have approved, for good or ill, that text.

Word processing, in effect if not in operation, is among the simplest kinds of computer software, but even word processing technology makes its epistemological impact felt. A text that starts as mere shadows on a screen and not as material impressions on a piece of paper, a text which remains at its core a series of electrical impulses, must necessarily be known differently than written or conventionally printed texts. The malleability of that electronic text must also give it a status closer to the spoken word than all other kinds of "written" texts, since it can be reconsidered and restructured in much the same way spoken text can (i.e., instantly, on the spot). Conversely, the printed version of a word processed text bears the look of a finished statement, thereby suggesting a greater finality than handwritten or even typewritten statements. In short, word processing technology spreads the notion of writing backward toward speech and forward toward print--and so makes the concept of writing less clear and less definitive. One can doodle on a word processor and yet end up with a very "official" looking document.

Like the rise of the daily newspaper or the coming of the paperback book, word processing will soon be absorbed in workaday life so thoroughly that its historical import may well be obscured by its familiarity. Yet, long after the operation of a word processor comes to seem a routine thing for many users, the epistemological impress of this technology will continue to be felt. As teachers, we would do well to include even the more basic computer operations like word processing in any assessment of what it will mean to know and to learn in an environment where, in many respects, we need not learn something because the computer we are using already knows it (like the word processor knows exactly where to put and how to order the characters we enter to create a document). As Vogeli observes, "the very way in which we think will be changed by a computer environment. What we consider to be knowledge will change. The way in which we acquire knowledge will change. Somehow we will need to include educational activities that teach students new epistemological principles" (88).

A fair amount of what has been written about computers thus far is, in one way or another, moralistic--arguing for them (mainly) or against them (occasionally) per se--whereas what is needed more is some speculation on their enormous epistemological, and so also social, effects. The questions of just how useful computers are (and can be) and who should or should not receive access to them are still very much open questions. What is no longer in doubt is that computers will be employed by large numbers of people and so will make a significant impact on life intellectually, socially, and economically--and that we should begin to assess that impact as it is taking place, rather than merely uncritically letting it happen.

5.6. Word Processing and the Idea of a 'Text'

In an essay entitled "Oral Culture and the Literate Mind," Walter Ong observes that "To the extent that you know really what writing is, you are not oral but literate" (135). The influence of word processing on the active side of writing--i.e., the construction of documents as opposed to their reading--was discussed in the previous section. Here we will take up the question of the reception of texts and just how word processing might change the manner in which we envision and receive them.

In several important ways the word processor will transform our notions of what constitutes a "text" and what constitutes "textuality." Print technology provided a ready-made teleology for texts; i.e., the printed version of a novel or law or catalogue came to be considered the definitive version, and was normally the only one consulted. As Eisenstein notes, print gave texts a certain enduring fixity that put them beyond immediate circumstances:

> Of all the new features introduced by the duplicative powers of print, preservation is possibly the most important. To appreciate its importance, we need to recall the conditions that prevailed before texts could be set in type. No manuscript, however useful as a reference guide, could be preserved for long without undergoing corruption by copyists, and even this sort of "preservation" rested precariously on the shifting demands of local elites and fluctuating incidence of trained scribal labor. (17)

Yet, no sooner did print technology give texts a literal fixity than there arose interpretative approaches that stressed the importance of context and connotative meaning in constructing a reading--ranging from the legal commonplace about the contrast between the "spirit" and "letter" of the law to the most elaborate hermeneutical systems of biblical interpretation.

Recent critical and interpretative trends have taken ideas of the legitimate variability of textual meaning a significant step further. Geoffrey Hartman, for instance, insists "There is no absolute knowledge but rather a textual infinite, an indeterminable web of texts or interpretations; and the fact that we discern periods or sentences or genres or individual outlines or unities of various kinds is somewhat like computing time" (202). Stanley Fish goes even further, arguing against any primal reality for the printed, material text:

> I challenged the self-sufficiency of the text by pointing out that its (apparently) spatial form belied the temporal dimension in which its meanings were actualized, and I argued that it was the developing shape of that actualization, rather than the static shape of the printed page, that should be the object of critical reception. (2)

While these views of the radical indeterminancy of the text may not be fully accepted outside the academic circles in which they circulate, such views are symptomatic of a growing philosophical distrust of texts as "containers" of meaning which readers need only "empty" intellectually through their interpretative gestures.

The whole web of electronic texts, from those produced by individuals at a word processing terminal to those chunks of a text "cut out" of huge electronic libraries and transmitted thousands of miles in a few moments, goes to reinforce such notions of the insubstantiality and incompleteness of a text in and of itself. Electronic texts can be instantly altered and rearranged--a literal act that is a loose correlative of what we do mentally when we read and interpret a text (emphasize some aspects, de-emphasize others, react to still others in terms of our own experiences, etc.)

In a more literal way, we shall soon be encountering some texts that exist only electronically on a primary level; things like large catalogues, bibliographies, and lists of information will not be printed at all, but simply entered onto a computer file and made available through it. Each user will receive the portion or portions of the text requested; if desired, these sections can be printed out--but as a single printed unit these texts may not exist anywhere. Moreover, such items are likely to be corrected and updated so regularly (say, once a month) that it would be meaningless to speak of a literal fixity for such texts: each time we consult them we may well be consulting a different text, the degree of difference being proportional to the intervals between our consultations. Electricity is a form of energy and such primally electronic texts might better be regarded as constant flows of information energy rather than as periodic fixings of available information (like a printed bibliography, that came out updated every three years, would be). Just what we catch from that informational flow will vary according to the exact moment we tap into it.

Our changing notion of what constitutes a "text"--how its essence may lie more in what interpretative gestures make it to be than simply in word imprinted on a page--is nowhere better exemplified than in considering what the advent of computers may do to the idea of a "textbook." For centuries, our educational system has been built on the availability in large numbers of textbooks, each copy of a textbook exactly similar in form and content. The uniformity of the books from which students learned a subject insured, indeed demanded, a certain agreement about what constituted the subject and the way and order in which its elements were to be learned. Pedagogy was structured to augment, expand, and apply the contents of these textbooks. In pedagogical practice the instructor often also had to compensate for a lack of certain attributes in the textbooks being used, most particularly material that would address the particular situation, needs, and abilities of his or her students and the specific requirements of a course and the sequence of which it was a part. With the coming of electronic texts, however, much more precise

alterations of instructional material (hardly "textbooks" in the old-fashioned sense) will become possible, will, indeed, become the norm. Consequently, teachers will probably find themselves doing much less teaching on a whole class basis and spending more time with individual students, helping them to shape an electronic instructional program in which material will be condensed or expanded as necessary, blended with the instructor's own material and exercises, and arranged somewhat differently for each student. Of course, "on the spot" clarifications, explanations, and even exercises will be necessary to compensate for ineffective or incomplete passages in the electronic educational material, or to deal with special circumstances as they arise. As an adjunct to all this, students will be using word processing programs for taking notes, reprinting material, reviewing it, and for writing essays, tests, and other assignments. With the increase in electronic education, software designers will need to restructure word processing programs, so they can be used in conjunction with educational programs, much in the manner students now use notebooks to summarize and outline material they are reading in a textbook. (One of the advantages of the computer system will be that students can create notes that incorporate, instead of just refer to, whole passages of instructional text.)

Of course, instructional software, like the textbooks being replaced, will still have to be written, even if its writing makes different demands on its writers than textbooks made on theirs. What will be produced will not be so much "a text" as simply "text" of the instructional variety. Very possibly this text will consist of a network of instructional matter, often created by many hands, from which each instructor will pick and choose sections, then structure and adapt them to his or her students. Similarly, when it comes time for students to show what they have learned about the material studied, they might not always write essays or essay exams, which imply a series of neat, finished statements, but rather produce a certain amount of "text" of their own, perhaps using their word processing programs to pull together, rework, expand, and condense things they have written over the term. The idea of periodic tests may give way to writing assignments that stretch over a whole term, with students adding, refining, and deleting written material on nearly a daily basis. Such an educational environment will also lead to changes in the way we view "subjects," very probably encouraging a more interdisciplinary approach, since the technology will allow for easy, rapid recombinations of material and so make it simpler to see parallels between fields. Often, in reviewing student writing, instructors will be looking at it less as a finished indication of a student's ability, and more in terms of its potential for development and refinement, since word processing technology makes even substantial revision and reworking possible, and pedagogically efficacious, within a short span. In all, the coming of word processing to the classroom will make the division between learning material and demonstrating how well the material has been learned less definite than it is at present. Instructors of the future may no longer, for example, design a course in which student performance will be judged by, say, two essays and two examinations, but rather only by a certain amount of satisfactory "text" each passing student must produce, the exact amount of that "text" and its precise content being something the teacher and individual student will work out over the course of the term.

This coming educational world may seem frighteningly amorphous, but as large numbers of students (and their teachers) work regularly with electronic

texts, the malleability of the text will become a more accepted and less troublesome notion, perhaps to the extent that many will view older, "fixed" forms (i.e., typewritten and printed texts) as frustratingly rigid. Certainly, most of the academic writers I know who have used word processing now think of typing a text as a primitive, slow way of going about its creation and revision--and as a method that commits them to producing a text in a limited number of "fixed" drafts.

Word processing technology is but the cutting edge of a computer revolution that will catch up students in the coming decades--whether it should be a great many or just a relatively privileged few will be one of the great educational debates of the remainder of the century. If it turns out to be the latter, our educational system will become further divided, with great numbers of students automatically excluded from the many positions and careers that will require, occasionally to no really useful end, a basic knowledge of and familiarity with computers. (It is having sufficient computer resources to provide the familiarity that is the expensive proposition.)

Because it is perhaps the most basic, and clearly the most serviceable, kind of computer technology, word processing is a demonstration of the typical advantages and limitations of working on and with a computer. It is also a demonstration of how even a relatively straightforward, essentially non-interactive computer technology is bound to lead to revisions of long-accepted notions in a print-oriented society; it is but the most ordinary example of how our conception of the "written" word, its transmission and reception have been and will be altered by the advent of electronic "knowledge."

A similarity of appearances can sometimes obscure vast changes, as it did in the first quarter century of printing when most printed documents looked very much like the hand-lettered documents professional scribes had been producing for centuries. The word processor's keyboard may look like a typewriter's and its printouts may resemble typed documents, but the word processor is not "just" a fancy typewriter, no more than the printing press was "just" a rapid scribe.

Notes to Chapter 5

1. The study to which Naiman refers was conducted by the National Institute of Occupational Safety and Health and is detailed in Potential Health Hazards of Video Display Terminals, published and distributed (free) by the U.S. Department of Health Publication office.

2. For now "computer literacy" seems to be defined only as the ability to know how to use a computer and not as the ability to know how to program it. However, as Reinecke points out, "Knowing how to use a computer without being able to program it might be compared with knowing how to read without having mastered how to write" (174). Still, being able to "read" gives one access to, if not power over, vast stores of information inaccessible to the wholly illiterate.

3. Erik Sandberg-Diment traces out some of the potential, and very possibly disconcerting, intellectual and stylistic consequences of writing in so tentative and impermanent a medium as electricity:

> Until rather recently, writing implied a certain permanence. While the physical act of alteration became easier over the centuries during the transition from quill pens and parchment to typewriters and rag paper, it still entailed some labor. One tended to think before letting the ink flow.
>
> Scribed electronically, letters are ethereal forms. They appear, disappear, reappear like some many apparitions called forth by the Oiuja board at the writer's fingertips. In such circumstances, the inscriber's thinking takes a new and different turn. The slow, deliberate, linear progression of structured logic, formerly used even when one was writing emotionally, gives way to a here-now, gone-in-a-second manner of composition. One's sense of continuity gives way to free-form thought. Dots of ideas resembling the dots making up the letters on the video screen drift in and out of one's consciousness, perchance to be captured, perchance to slip away unrecorded. (C5)

APPENDIX A: SAMPLE STUDENT DIRECTION SHEET

As mentioned in the second chapter, it is very important to distribute to students a simplified, clear list of directions for operation of the word processing program they will be using. Such direction sheets should take students through the basic steps for creating a document on their own and should provide a reference to which they can return for essential commands.

The general direction sheet printed here was first developed by my colleague, Marian Arkin, and modified by both of us over four terms, until it is now a simple, effective document that informs students in our program just enough of--and no more than--what they need to know in order to use WordStar to write. Obviously teachers with different populations, or teaching subjects other than writing, would have to construct such a document somewhat differently. Besides clarity, the key to these direction sheets is simplicity: remember that the students are not being taught word processing, but only enough about word processing to enable them to write and revise the written assignments for a particular course.

GENERAL DIRECTIONS FOR USE OF THE WORDSTAR PROGRAM

Dear Student:

Following is a list of things you need to know in order to work with the computer for this course. Please bring this sheet with you to class until you feel you know the material on it by heart. Don't panic if you don't quite understand what you're doing. It takes some time, but eventually you will feel quite at ease.

You are using a word processing "program" on a microcomputer. The program is called WordStar. WordStar is a good program for writers like us because it is easy to do such things as correct, insert material, delete material, move material around, store your written work, and much more. Here's what you should do each time you come to the lab:

1. When you sit down, there should be a green square and an A on the screen. (If there isn't, call me over.) Type the letters WSH and press the return key. This will allow you to use the WordStar program.

2. Something called the "NO FILE MENU" will appear on the screen. Like a menu in a restaurant, this menu lists what the program can give you; in this case the machine can do things for you, if you know the directions. In computer language, directions are called commands. The command D will allow you to write or edit an essay, so type the letter D next.

3. Read the instructions that appear on the screen. They pertain to the rules for naming a file. Normally I will give you the name of your file, so that everyone will have the same assignment titles. Type the name of the file you want to open or return to.

4. Now you want to set the spacing. Since it will be easier to read your essays if they are triple-spaced, type Control (CTRL) OS. Then type the number 3. (Do not hold the CTRL key when you type this number.)

5. Type the title of your essay on the first line; then center it by pressing CTRL OC. Then press RETURN.

6. Under the title write your name and center it by pressing CTRL OC. Press RETURN and then TAB once (to indent).

7. Now you are ready to write. Pretend this is a normal typewriter--but YOU MUST REMEMBER ONE IMPORTANT DIFFERENCE. The cursor, that bright green square, tells you where you are. To move the cursor, you do not press the space bar because it will literally create a space. THE WAY TO MOVE THE CURSOR IS TO TYPE CTRL and either E, D, S, X, A, or F. Think of it as a triangle:

Ctrl E moves the cursor up.
Ctrl D moves the cursor one space to the right.
Ctrl X moves the cursor one line down.

Ctrl S moves the cursor one space to the left.
Ctrl A moves the cursor one word to the left.
Ctrl F moves the cursor one word to the right.

Play around with the cursor for a while. It is, perhaps, the most difficult concept for you to learn on the machine.

8. To insert a letter, word, or sentence, just move the cursor to where you want to make the insertion and begin typing.

9. The machine will make your righthand margin for you automatically, so DO NOT PRESS RETURN unless you want to begin a new paragraph.

10. Those of you who are typists, watch out for the proximity of the CTRL switch to the SHIFT key.

11. Finally, when you are done, press CTRL KD. This will save your work so that it does not get erased (by your mistake or machine error).

12. Then press P. The machine will ask your NAME OF FILE TO PRINT? Type the name of the file you want to print. Then press RETURN, then ESC. Set the printer on the letter corresponding to your machine (A, B, C, or D) and your essay will be printed out.

ERASING

1. To erase a letter, move the cursor over it and type CTRL G.

2. To erase a word, move the cursor over the first letter and type CTRL T.

3. To erase a whole line, put the cursor anywhere on the line and type CTRL Y.

4. Also, you can use the RUB OUT key to erase the letter or space to the left of the cursor.

OTHER USEFUL COMMANDS

1. To underline, type CTRL PS before and after the word or words you want underlined. Be sure to put in the second CTRL PS or the machine will underline the rest of your essay.

2. To move forward a whole screen in your essay, type CTRL C. To move backward a whole screen, type CTRL R.

3. To move to the beginning of your essay, type CTRL QR. To move to the end of your essay, type CTRL QC.

PERIODIC SAVINGS

To guard against machine failure, you should save what you have typed every 10-15 minutes. Just type CTRL KS and then, after the save, type CTRL QP to return to exactly where you were.

REFORMATTING

1. Reshaping and realigning your text after you make insertions and deletions is done paragraph-by-paragraph.

2. Move the cursor to the head of the paragraph and press CTRL B. One of two things will then happen: either the whole paragraph will be reformed and the cursor will appear on the line beneath it, or the cursor will pause at the end of a line within the paragraph and ask if you want to hyphenate. Since we are not using this hyphenation feature, simply type CTRL B again--and keep doing so until the cursor moves to the end of the paragraph.

3. If the whole paragraph does not reform, it is probably because you have accidentally put one or more carriage returns in it. Look for the symbols on the right side of the screen, move the cursor to the end of the line on which they appear and type CTRL G to eliminate them. Then move the cursor to the head of paragraph and press CTRL B again.

APPENDIX B: CLASS DIARIES

Working with word processing programs, both individually and pedagogically, can fairly quickly become a smooth, semiautomatic procedure, with the key issues, problems, frustrations, and discoveries of the initiation period soon forgotten. This appendix contains a class-by-class diary that I kept during the first two terms our microcomputer/writing program was in operation. A number of visitors to our program, particularly faculty and administrators, faced with the imminent introduction of microcomputers into their writing program, have found this diary a useful complement to the more organized and neater description of our program given by my colleagues and myself now that we are "veterans." Precisely because the diary mirrors early doubts, notable surprises, evident vacillations, and important revelations in launching a writing/computer program, it necessarily gives a more realistic sense of that process than a summation of it after the fact.

This diary covers my experiences with word processing in English 099 and English 101 during the winter and spring 1983 quarter terms. English 099--our third and final developmental writing course at LaGuardia Community College and the one into which the largest number of entering students place--is the core of our program and the most frequently taught course. Regularly, English 099 meets for four 70-minute periods a week over a ten-week quarter; in our program, it meets for two double periods a week--the first 70 minutes in a classroom, and the second 70 minutes in the microcomputer lab.

After passing English 099, students take English 101 (Composition I), the standard introductory composition course required of all LaGuardia students. Usually, it meets for three class periods a week; in our program, there is one double period a week, equally divided between the classroom and the microcomputer lab, and a second lab session.

Following English 101, LaGuardia students, depending on the requirements of their major, take either Writing for Business or Composition II: Writing Through Literature. Both courses meet three times a week; in our program, the form meets twice a week in the computer lab, and the latter once a week.

It might be added, as a means of putting the diary in perspective, that while we had three full months to plan the program, our computers arrived only two weeks before classes were to begin--and so we barely had time to learn the basics of word processing ourselves before we had to teach them to writing students.

BASIC WRITING DIARY (ENG 099)

1-4-83

The first reaction of students to the computer is extreme interest--they are fascinated by it and want to know more than we had planned to tell them. They also seem to have a genuine belief that working with the microcomputer will aid them significantly in overcoming their writing problems. Emotionally--at least as regards the college experience and the writing courses that are part of it--they show a range of attitudes. One student, now in her fifth quarter, admitted that college has not been "too great" for her--but another, in her second term, professed to liking LaGuardia and feeling happy about writing (even though she did not pass the course the first time through).

One student thought their essays were to be corrected by computer, and was, I think, quite relieved to discover that matters had not gotten that mechanical.

Students are clearly pleased to be part of a special experimental program--they feel the college is doing something extra for them. At first impression, the students' attitude could hardly be better; they are cooperative, enthusiastic, and quite verbal. They are also aware that they have major writing problems that need to be remedied. Moreover, they indicate that their writing problems are by no means indicative of their general level of intelligence and ability to do college-level work.

Diagnostic Essays: Essays from the first students indicate a similarity of problems: spelling (severe in all but one case, full of inversions and oddly phonetic spellings), missed apostrophes, omissions, capitalization errors, run-on (really more run-together) sentences, inappropriate word choice, and subject-verb agreement (which often seemed a variation of spelling errors). Conversely, these students wrote "well" in the sense they clearly had something to say and a fair idea of how to go about saying it. As a group, they probably write better than the average 099 student in terms of content and noticeably worse in terms of mechanics, albeit their mechanical errors are of a very specific and limited sort (which makes it more likely they can be concentrated on and eliminated more easily than more "scattered" errors).

One other point: several of the students demonstrated oddities in their handwriting, one's being severely cramped and barely legible, another's exaggeratedly large.

1-6-83

The students, but one, are quite extraordinarily pleased to be putting their first small essays on the computer. Correcting mistakes, normally a chore, now seems almost a game. They are eager to correct errors the moment they notice them, not even wanting to finish typing the essay first. The appearance of their work on the screen obviously allows them to see what they have done wrong or missed more easily. There are limitations, of course--e.g., one student saw that "brus" was wrong and, using the dictionary, corrected it to "bruise," but neglected to notice that the context called for the plural form. Indentation, a process to which many 099 students are

indifferent, is something students now notice and want to practice. As for instruction on the computer, the next steps should probably be:

1) review of cursor movements (with practice in correction on individual basis)

2) reemphasis on avoiding "return" if reformatting after corrections is to be efficient

3) explaining and practicing the different correctional uses of "rub out" and Ctrl G.

One new student had a completely different reaction to working with the computer--she saw it as exposing her "errors" for all to see and, apparently, scorn. After a lengthy discussion with me and the project director, it was determined that this student, who was in a genuinely tearful state at the prospect of having to work on the computer, would do better in a regular section of the course. It is interesting, though perhaps not ultimately enlightening, that her reaction was so diametrically opposed to that of all her classmates.

1-11-83

An act of God, or at least of the water department, truncates today's class. A water main break in the morning forces the school to close at 10 a.m. Unfortunately, the class session is just at a key point--after some "warm-up" exercises using verbal and written descriptions with photographs, the class is two minutes into a narrative free-writing exercise (working from a photograph) when the evacuation order comes. I take it as a good sign that a number of the students are upset at being interrupted in their work.

1-13-83

Although there are still some (new) students who must learn the use of the microcomputer from scratch, today's is the first session where most of the students are spending most of their time actually writing on the machine, rather than essentially learning its procedures. If anything, the difference in speed--with some difference in competence--between those who can type and those who cannot is exacerbated when working on the microcomputer. A real problem for students who do not know how to type is unfamiliarity with the arrangement of the keyboard--they spend a great amount of time looking for the letters they want to hit.

Most surprisingly, the student who left class last week out of dread of the machine returned today--she seems reconciled to working on the machine and, if not as enthusiastic as other students, at least seems to recognize that writing inevitably means that others will--must--see your work.

At this point, students are at best "even" in terms of speed--they are not writing any faster than they would by longhand or on a typewriter. We shall see from the expansion and correction of their first drafts the potential usefulness of having a clear printout to make corrections on.

There seems to be correlation between machine use and content--or, more accurately, between reflection on how to use the machine and reflection on the content of what they are writing. Students are starting to ask intelligent questions about the computer functions--doing so appears to be leading them to ask similar questions about what they are writing. We might further consider and look to make use of this interplay between the "how" and "what" of their writing.

1-18-83

Today is a real advance--students are putting their first full-length essay on the machine. Several work well and quickly, others more slowly. The time variable is obviously crucial--students who can type finish in about half the time of those who cannot. Some adjust by writing half-length (175-200 words) essays. Others simply write as much as they can. We will have to create useful work for the speedier students; they can do extra revisions for awhile, but ultimately they might require extra assignments, lest they become too jaded correcting the same essay.

Absolutely essential: more practice hours are necessary for students. Even if my students are allowed to remain in the lab for the following class period hour, four students will still not have any practice time.

Several students are chary about the battery of tests they are being given: some have been tested several times before and are suspicious of having this process "done" on them yet again.

One student had to withdraw today because she is taking a medical leave of absence. She was very firm about wanting to be in a microcomputer writing class again next term and was pleased when I assured her that this would be arranged--a good sign.

Although prone to sudden frustrations, the students, overall, seem more comfortable writing on the machines than we might have suspected given their relatively small degree of knowledge about and experience with the word processing apparatus.

1-20-83

Today is another advance--students make their revisions (and learn the revising process as they go) and, if necessary, expand their essays. Of course, some time is taken up by fixing "errors" created as they format (e.g., single-spacing groups of lines in the middle of double-spaced essays). Almost all essays are up to the 300-word lower limit, and a number exceed it. One problem: students still tend not to make all the corrections they have marked on their first drafts. Also, they are a bit disturbed by the very scattered look of the paragraphs as they revise and insert; they want to reformat each time they change a line length. Conversely, they have come to believe in the machine's capacity to keep all they have written and no longer worry about their text disappearing up, down, left, or right.

Two issues:

1) Tonight I will review these second drafts, but I will need to be suggestive rather than proscriptive in my comments, so the final version they will produce in the next class will still be their work and gradeable as such.

2) We might have some "separation anxiety" on Tuesday--i.e., given the inherent capacities of the machine (its "correctability" fixation), some students might want to keep working and reworking the same essay and not be willing to move on to a new project.

1-25-83

Today's is an absolutely exhausting session. One must remember that working with students in this context demands greater individual attention than an instructor can necessarily always provide. Our collective decision to limit future classes for learning disabled students in the microcomputer lab to twelve is essential.

Students are still easily frustrated by what they see as irrational responses on the part of their machines--and they often respond in kind. Still, the last version of the "My Childhood" essay they are producing clearly demonstrates that they are learning rapidly--the revision movements are complicated and they are making 80-90% of them on their own. A source of frustration, though, is the fact that they are missing some of the corrections they have marked on earlier drafts. (In a sense, they think this unfair, since they have already noted and corrected as necessary.)

It will be crucial to grade these students holistically. Many are already worried and talking about the exit exam--and they want to be sure they do enough writing (i.e., number of essays) to give them the best chance. We assure them the exit procedure is different in this version of the course--an assertion they will believe as long as they seem to be moving toward passing the course.

Had to work with students individually for two hours after class--some need more practice than others <u>with someone standing by to help</u>.

2-1-83

Results of the first essay, in its final (third) form, are very heartening: ten of eleven students pass, with eight of those eleven absolutely clear passes (C+ and above). With the two "marginal" essays--a bit short and 5-6 grammatical errors--I gave "D" grades, since it is very difficult not to give some reward and recognition for the very considerable improvement these two students showed from version to version. Students are much cheered by their performance--they really seem to see some definite possibility of passing--and eagerly apply themselves to today's writing task, revising and expanding the essay they wrote in the self-paced exercise of the last class. One conclusion, aside from the very clear improvement in their writing performance on the machine, is that they have come to understand the long and careful process that revision is. Another point: several students, including one whose last essay was short, are writing more in this second

essay. One seems to have moved up to about 500 words and is working out an essay full of very good, realistic dialogue.

This class is beginning to take on heavy secular overtones of "faith." The students are coming to believe--in the development of their abilities, in the great capacity of the machine to help them, in the whole writing process. There are still brief moments of "heresy" when the machine does not function as they expect, but these are becoming infrequent.

One student continues to be a problem--or rather, to be fair to him, continues not to find the class a valid, functional realm for him. His psychological problems are more extreme and disruptive than those of any other student. He has difficulty focussing on and completing a task, often erases his work after completing two or three sentences, and rarely gets more than 100 words for his day's work. Clearly, he needs a different kind of class or, more likely, a one-to-one tutorial. His behavior is sometimes disturbing to other students--e.g., he insisted on the classroom door being open today, although many students wanted it closed for quiet. Unfortunately, there seems to be no way of balancing his strong needs with the needs of the other ten students. Must look into other pedagogical possibilities for him.

2-3-83

Class goes very well today--students, save one (to be truthful), are extremely interested when I mention some aspects of the evolution of the English language and want to know more. A twenty-minute mini-lecture results with, I think, profit to them. Although I've spoken to classes before on this topic, I have never found the interest so widespread as in this case. It may be that the close kind of work we are doing on the structure of their essays has produced a keener interest in language as a structure which needs to be studied before it can be fully utilized or manipulated.

Lab session today goes much more quietly than past sessions. Students are working on penultimate drafts of their essays and appear much surer and more confident about working with corrections on the machines. For the first time, their demands for attention to their work do not wear me out. In future sessions, I will, doubtless, have to suppress my urge to interfere, my urge to make corrections for the students before they can make them for themselves.

Two points for future planning:

1) It is absolutely imperative that these students, a majority of whom will likely pass 099, have as much access to the microcomputer lab as possible while taking 101. They will need to write and rewrite and rewrite again, if they are to produce passing essays in 101. Denied this access, they will likely regress to the level at which they began in 099. It would also be useful if they could get some extra assistance from faculty in 101, so the transition to working essentially "alone" will not be so abrupt. Ideally, each will be able to take 101 in the "microcomputer mode."

2) If not, each student who passes 099 should be tracked into 101 and his/her instructor notified of the special 099 course the student took and how, very probably, the student will ask for and, absolutely certainly, need

extra help with his/her writing. Ideally, we could set up a special section of 101 for this group--it would, however, have to wait until the fall quarter.

Miscellaneous point: the first drafts of the second essay seem, on quick inspection, no better in form or content than the first drafts of the first essay. But then, the students may feel, and probably rightly, that working on the microcomputer allows them the chance to "get it all down" first and worry about corrections afterwards.

2-8-83

Guided writing exercise (shifting a passage to the past tense): students have a great deal more difficulty with this than average 099 students. Many instances are missed--apparently because many students have difficulty approaching the task holistically, recognizing that the entire passage, and not just selected verbs, needs to be shifted. A number of students end up putting verbs into the perfect tenses, including one student who put several verbs into the future perfect. When I explain this tense to the class, students are much amused, struck by the unlikelihood of ever being in a situation where this tense is called for. Their futures clearly seem more indefinite than the certainty called forth by this tense.

In lab, students work very diligently to produce final versions of their essays. If nothing else, working on the microcomputer--perhaps because it produces so definite a printed product--has made them willing to be much more careful about their writing.

2-10-83

Today a videotape--"Before the First Word," on prewriting an essay on a neighborhood--stimulates them. These students, not surprisingly, are very visual, able to recall small, exact details from the tape. From this we build to visions of their neighborhood (expressed verbally). A short summary of descriptive writing via concentration on as many of the five kinds of images as possible: works well.

First half of lab period devoted to dictation--this exercise always proves revealing for these students, indicating just how much they mishear, how many letter reversals, missed endings, and misspellings (some genuinely bizarre) they have. They are not able to see all of these even when they compare their printed copy with a correct copy. In short, these students are struggling with extraordinary writing problems and their coherent, generally correct final versions of essays represent a real triumph of will over considerable difficulties.

Today's free writing on the machine--unlike the first attempt--works well. Students were able to loosen up much more and do real free writing (about their neighborhoods), rather than, as they did earlier, just doing a first draft. This free writing is full of specific, evocative details. The developing essays on their neighborhoods seem very well suited to these students, especially since they originated from sensory observations.

2-15-83

The essay writing on "my neighborhood" seems to be going well, but we will need to work more thoroughly on showing students how to integrate material from their free writing and outlines into the draft of a paper. Details evolved are very good, very sharp--e.g., "the sour smell from a pile of used 'Pampers'," the picture of "winos" hanging out on the steps of the corner church, the quiet that envelops one on turning into one's block from a noisy, dirty street.

Students are very enthusiastic when I mention the possibility of taking Eng 101 in this "microcomputer format," even though they might have to wait until the fall term to do so. As one put it, it "seems a shame to waste" all the microcomputer skills they have learned by thrusting them into an ordinary section of the class.

2-17-83

Students are very pleased when I tell them we have reworked our program schedule and that they will be able to take Eng 101 using the microcomputer next term. Several are even willing to make substantial schedule readjustments to do so.

Might use John McPhee's Oranges, as in past, for these students to give them a holistic sense of the prose writing act.

2-22-83

Students seemed to have reached a still point--i.e., they are letting a standard number of errors, chiefly letter reversals and subject-verb agreement mistakes, into the first draft of their essays, trusting they will notice and be able to correct them as they read printouts (only a few are caught when the text is on the screen). It would seem that working with the microcomputers does not so much allow them to internalize correct spelling, grammar, and syntax (the ideal situation) as it allows them to "make and remake" their work until most of the errors are eliminated.

2-24-83

The "classmate interview" exercise works very well--these are, in the main, gregarious students (although one described herself, tongue in cheek, as "anti-sociable" and they take to an interpersonal assignment like this. Our concern here is that students might have difficulty with this essay because, for the first time, we are not giving them a loose structure for their essay but rather insisting that they determine the shape of the essay from the shape of the raw material they have gathered in the interviews.

Student evaluations done today--whatever the results, generally positive I assume, will be compromised by the fact that the form is so very badly suited to the realities of any 099, or other writing, class.

3-1-83

Much madness is greatest sense?? Today the students are rather manic, perhaps because they are feeling comfortable and confident about working on the machines, perhaps because they are a bit bored after having "conquered" the basic operation of the microcomputer. Still, the drafts show the students are writing better and better--but the "better" refers chiefly to content. They seem a bit reluctant about going over and over their essays for grammatical and technical errors. Having their "subjects" sitting next to them gives a good sense of how one needs to go back to retrieve more information and refine/sharpen statements to exactitude. Anxious queries again about the certainty of a 101 running next term.

3-3-83

Students still very keen about taking 101 in this format; hope we can work out independent study credit for the extra hour. All allusions to this term's course (in the essays they wrote on each other) are quite positive.

3-8-83

Things a bit hectic about trying to work out an essay in two class periods, this being the first. About half the students, because they missed a day somewhere along the line, still have one (or more) essays to put into final draft form--all begins to run together in terms of time for them, yet they seem able to hold on and continue to write/revise competently. It is absolutely vital that students have free access to the lab many hours a week; otherwise, they will simply not be able to complete their work. This issue will become more acute when we start on 101--issue must be pushed. It looks like 10 of 11 will pass, if all ten make up their late work by the end of the week. Content of some essays is really remarkable.

3-10-83

Final essays prove quite good--except for the one student who stood no chance of passing. The lowest mark is a B, with most essays B+ or A. (Marian marked my students' essays and I hers.) This is a considerable accomplishment. Students are extremely pleased with themselves, surely an indication of how useful a bit of success is for basic writing students--and how a little success breeds more success breeds more success. Many of these students learned to write better in one term (often after repeated failures in 099) than they ever thought they could. Our emphasis on the positive qualities of their writing, rather than on error count, clearly enabled them to build on their successes rather than, Sisyphus-like, having to spend the whole term pushing the heavy rock of their "mistakes" up the slope.

3-15-83

We come to our quarter's conclusion--and a quite satisfactory one at that. The students worked very hard and very productively on their writing and deserved their success. Both they and I are very pleased that they will be able to move on to 101 next term. Not only do they need to build on the improvement they made this term, but they also need the special care and attention this kind of class gives them. Without special provisions, 101

would be a disaster for them--they would, I fear, revert to their bad writing habits and quickly be overcome by the work. If our success this term was considerable, we must not forget that these students received much extra attention, spent additional time in the course, and had the aid of the microcomputers. I suspect any group of students working under such circumstances would do markedly better. Still, the learning disabled students in the class could hardly "do" at all without this kind of program. The simple fact of the matter is that these are more expensive students to teach because they cannot be taught without special programs. However, seeing such a program work so well makes one aware how "cost-effective" such programs are in terms of human resources.

COMPOSITION I DIARY (ENG 101)

4-5-83

Much enthusiasm from returning students--they willingly testify to the new students (four, with twelve returnees--sixteen being rather an unwieldly number in the lab, very tiring, barely manageable). All the returnees seem to have retained their ability to use the machine easily and efficiently, forgetting only the less vital commands (like centering). On the other hand, a free-writing exercise shows that they continue on the first draft to make the same "old" errors: letter-reversals, missed words, skewed syntax. Now, however, they can correct them as they work from draft to draft--and know how to do so. Most of them will probably have to live with these errors, will probably continue to make them. What the computer can really do for them is to give them the ability to spot and correct such errors.

4-7-83

Exercise with a narrative passage from Oranges works very well. These students are fascinated by the structure of the piece, and perform an intelligent analysis to explore and understand it. They get very directly to the narrative "work" of McPhee's paragraphs.

Time a bit off in the lab session--i.e., I did not leave a full period for them to do a first draft. They clearly needed at least 70 minutes to draft a 400-word essay. Exact time will have to be made in next class, since it is crucial that students go through the whole draft-revising-rewriting process. Future exercise will have to be cancelled to get enough writing time in. Although students will be finishing essays outside class time (in their practice hour), it is vital that they have a good, "finished" penultimate draft from which to work.

New students seem to be fitting into class very smoothly. The veterans take well to being mentors. Registration procedures for future classes (both in terms of four-hour 101 and a practice hour) will have to be refined: nearly half my students have to arrange another hour (than Wednesday at 10:40) for their practice period, and several are having real difficulties because of heavy use of the lab.

4-12-83

Adjustments to schedule are in order--more time (perhaps one more lab period) will need to be devoted to completing essays. We will not be able to do the exercises planned, but the sacrifice will undoubtedly help the writing-revising-rewriting pattern become more firmly established.

A few problems with length--students functioned well at the circa 300-400 word length of last term's essays and seem instinctively to be envisioning and arranging their essays on that model even though asked to write 500 words. I don't anticipate this as a major problem; in fact, it can be turned to an advantage by working on methods of expanding essays (where to add sentences, what else can and should be said, etc.).

New students doing fine. I suspect that as long as no more than 25% of a class is new students they can learn the microcomputer as they go and more or less keep pace with the veteran students.

4-14-83

Having a few problems with time on this first essay--specifically: only three students handed in final versions of essay #1 today as scheduled. The other thirteen still need some lab time to finish the work--and they have allotted same, a good thing. Only problem is that they might not finish #1 before they begin on #2. I will have to watch the overall pattern of time for the term closely, to insure that all students have sufficient time to finish all the essays, including what could be the somewhat problematical research assignment.

The three essays handed in look good. Some other students are having a bit of difficulty generating essays of required length. Ironically, a few of them found it easier to generate strong content when the course--i.e., 099--centered less on content than coherence.

4-19-83

More essays filtering in; ones received so far seem fine. Obviously, with a variable lab hour built in, the schedule of exactly when students hand each essay in will have to be more flexible. My attempt to have them take more responsibility for their completed work ("Only hand me your essay when you are satisfied with it") seems to be effective--only difficulty might be time pressures toward the latter part of the term.

Descriptive outlining (from free writing on the microcomputer) works well. To approximate a regular 101 class, more of the typical, regular 101 classroom assignments will have to be reworked into lab format. It probably is that 099 adapts itself better to the 2-classroom/2-lab format we are using for these classes--but to get any real use from working on the microcomputers, 101 students need to spend two full 70-minute periods per week plus an open lab hour working on the machines. They also must have two full classroom periods. In short, to work, this kind of 101 class needs to be more fulsome and intense than regular 101s. I suspect the results will prove the extra time and effort cost-effective, to use a monetary measure, and very rewarding, to add a human one.

4-25-83

Much valuable progress today--it looks like most of the class, working in the lab period tomorrow, will be able to finish essay #2 on Thursday and be on schedule. One of the advantages of working on the microcomputers three periods a week is that students pick up speed after a few weeks and can complete assignments more and more quickly. Also, for 101, where essays are normally done (all but one or two) at home, it keeps students on a more regular schedule to have to write three days a week on the machine, and thus helps insure they will complete all the assignments by the end of the term.

Classroom period proves a bit sluggish for the first 20-30 minutes today (until I can get them fully involved in a group exercise)--suspect they were anxious to get to the machines and continue work on their essays. Still, they did finally, I think, get the point of today's classroom exercise: on the difference in speaking to and writing for an audience.

Very anxious to have the WPA evaluators visit the class and particularly to have them talk privately to the students--evaluators will, I suspect, be quite amazed by the very enthusiastic attitudes of students toward this learning experience. Also, I want it on record how valuable and positive this experience has been, especially for students who have had few positive experiences with learning up to now.

4-28-83

Class is very spirited today--students are taking more responsibility and appreciate the fact that they decide when an assignment is finished and when they want it marked.

I realize now that the research assignment will have to be greatly modified to fit the format of the microcomputer class.

5-17-83

Things rather scattered of late--students working at very different paces. Had to let one student who had finished (and very well) her latest essay ahead of (the revised) schedule, go to the library during lab period, so she could begin work on her research assignment. As for that assignment, it is now clear that I will have to modify the classroom-lab format for it even more than originally planned. I've asked the students to complete research by a week from today, so they can begin putting the material for this rather complexly organized piece on the machine. The specifications of this assignment--not exhausting but very precise--are likely to cause these students difficulties.

Class on logical writing--usually one of my best and most exciting lessons in 101--was decidedly tepid today. The students, unlike regular 101 students, simply did not see logical flaws in the examples I gave them and needed a great deal of explanation before they could understand what was logically wrong.

The research assignment, I now see, will be a real test for them: if they can complete this as directed they will have made another very significant

advance, one that will be very valuable for all their subsequent upper-level work.

5-19-83

Students are rather more interested than one would have expected in today's lesson--the standard, and inevitably somewhat boring (though necessary), lesson on footnoting and bibliography. Clearly this kind of ordering appeals to them. They ask a host of questions, all indicating that they are following what I say very closely and genuinely imagining alternate situations.

This week's two class sessions demonstrate--clearly and again--that these students do not follow normal patterns. The class on logical analysis and logical writing (normally one of the term's best and most exciting) fell rather flat (logic is very difficult for them), but the mechanical and precise imposition of order via footnotes and bibliographies catches their fancy.

Several students starting early on research assignment today--my decision to revamp format to include more lab work seems wise. I believe that they all, given sufficient time to do drafts of the complex (if not particularly long) will now be able to complete it.

Thought: the logistics of working with the microcomputers inevitably means assignments take longer to complete and not quite as many can be given as in a regular class; yet, this diminution in quantity seems more than amply compensated by increase in quality. Perhaps there is a general lesson here: fewer assignments and more concentration (i.e., revision and revision) on each.

5-24-83

Several students are progressing very rapidly with the research assignment. One should finish it two classes ahead of time and another, one class ahead. Conversely, some students are behind, not beginning a draft in the lab period today, as scheduled. More than ever, 101 seems to be running on a variety of time schedules. Those students who are used to taking responsibility are able to extend that ability to doing research and incorporating it in an assignment. Others appear to be waiting for me to give them even more explicit directions on how to progress than the already detailed ones I have given. Still, with all my "shepherding," all but one or two should finish all their assignments by the end of the quarter.

The most competent student in the class is now doing all her written work, including two other term papers, on the microcomputer. She is very good at organizing her work on her own and taking advantage of the lab whenever there is a free terminal at which she can work.

5-26-83

Those students doing very well on the research assignment will finish ahead of schedule. All now seem involved and have a clear idea of precisely what they need to do. A few, however, have not yet completed their research and so cannot utilize the lab period to start putting drafts on the machine.

Reluctantly, I send them down to the library for the period, emphasizing (again) that it is their responsibility to complete all work by the end of the term (two weeks hence)--they do seem to appreciate this fact. I worry, though, that several will not have sufficient time to be able to get the research they have done into the exact format called for in the assignment. Most are, however, getting rather excited about doing research. I suspect that they are repeating in doing research the pattern they went through in learning to operate the machine--namely, what at first appeared a mysterious and overwhelming task, soon proved very much easier, more doable and even fun.

Withal, we shall have to restructure 101 more significantly for its second run-through than we had to restructure 099.

5-31-83

Work on the research paper--which raises students to a new level of responsibility for their own work--proves very difficult for several and has me hopping about like mad in the lab. But as they learn they can do it, learn that they can understand and adapt to the specific format, they are pleased. It proves a new challenge within reason. Many of these students have been conditioned in passivity by their former pedagogical environments, a condition abetted by their use of passivity as a defense against failure. They must gradually be brought to activity.

6-2-83

In my end is my beginning--very, very hectic today, with much racing from student to student. These last few classes are "mop-ups" for the term. Almost all students are at different points in the research assignment and working hard to assemble a coherent whole. (The decision to have them work on the three parts of it as separate files does mitigate the chaos a bit, making it simpler for them to work on whatever part they need to revise/expand.) Next week many will have to spend extra hours in the lab to finish the assignment, since we are scheduled for a final essay (an argument paper) in the last two class periods. They are, after travail, doing well as regards researching and putting information into a coherent whole.

6-7-83

As happened at the end of last term, the students are "rallying" this last week--completing overdue assignments by staying extra hours in the lab. We could hardly squeeze in all from the 10:40 class who wanted to stay to work in the noontime period. Whatever else these microcomputer courses have done, they have clearly led the students to take more responsibility for their work. Unlike most 101 classes, where students typically vanish during the last week if they have a significant amount of work to catch up on, these students are very willing to put in the necessary extra time to complete all the work. And most are completing it in more than just passable fashion.

I do worry, more than ever, about "separation" anxiety when the students no longer have the machines available and, more specifically, when they no longer have the enforced discipline of only being able to work at the machines for a limited number of hours each week. Still, these machines, or their near like, are the writing implements of the future, and most probably these

students will soon again encounter them, and be all the further ahead because they already know how to work them.

6-9-83

Things end well--students do concentrate on their final essay, an argument paper on "Can Racism Basically Be Eliminated from American Society?" Results are generally quite good, especially since they are rushed and a number are running behind.

One student, who had three assignments to put into final form, has not shown up for the last three classes--his absence underscores the fact that four other students who were in roughly the same position did show up and did complete the work with passing grades.

Students do quite well in toto: most get marks in the B range. There is, however, only one A out of sixteen students. If I were to guess why this slight "decline" (four students who got A's last term got B's this), I would attribute it chiefly to the larger class size. Although sixteen students are manageable in 101, I think the optimum class size is 10-12, unless there is additional help. At this size it is possible to work individually with each student during each lab period; with a bit more guidance, encouragement, and prodding, I think at least some of the six students in the B+ range would have made it to A. Another reason for the slight decline may be that 101 simply does not represent the kind of "breakthrough" into competent writing that 099 does--i.e., these students have discovered that they can write clearly, can pass writing courses. Having done so, and often quite handily in 099, they are more confident but less "thrilled" about the work in 101.

Works Cited

Akst, Geoffrey, ed. Microcomputers and Basic Skills in College: Applications in Reading, Writing, English as a Second Language, and Mathematics. New York: Instructional Resource Center, CUNY Office of Academic Affairs, 1984.

Appleby, Bruce C. "Computers and Composition: An Overview." Focus 102-110.

Arkin, Marian, et al. "Word Processing: A Catalyst for the Basic Writing Student." Akst 24-26.

Arms, Valerie. "Creating and Recreating." College Composition and Communication 34.3 (Oct. 1983): 355-358.

Ayer, A. J. "Let Us Calculate." New York Review of Books 1 March 1984: 15-16.

Baum, Joan. Computers in the English Class: With Particular Attention to The City University of New York. CUNY Research Monograph Series: Report No. 6. New York: Instructional Resource Center, CUNY Office of Academic Affairs, 1983.

Bean, John C. "Computerized Word Processing as an Aid to Revision." College Composition and Communication 34.2 (May 1983): 146-148.

Bernstein, Basil. "Social Class, Language and Socialization." Language and Social Context. Ed. Pier Paolo Giglioli. New York: Penguin, 1972. 158-178.

Bourque, Joseph H. "Understanding and Evaluating: The Humanist as Computer Specialist." College English 45.1 (January 1983): 67-73.

Brown, Dale. "Learning Disabilities." Fact Sheet. Reston, VA: ERIC Clearinghouse on Handicapped and Gifted Children, 1980.

Bruffee, Kenneth A. A Short Course in Writing. 2nd ed. Cambridge, MA: Winthrop, 1980.

Burns, Hugh. "Computer-Assisted Prewriting Activities: Harmonics for Invention." Lawlor 19-29.

Burns, Hugh L. and George H. Culp. "Stimulating Invention Through Computer-Assisted Instruction." Educational Technology 20 (August 1980): 5-10.

Christ, Frank L. "Techtalk: Using a Computer for Instruction." Journal of Remedial and Developmental Education 7.1 (Fall 1983): 26-27.

Collier, Richard M. "The Word Processor and Revision Strategies." College Composition and Communication 34.2 (May 1983): 149-155.

Daiute, Collete A. "The Computer As Stylus and Audience." College Composition and Communication 34.2 (May 1983): 134-145.

Eco, Umberto. A Theory of Semiotics. Bloomington: Indiana UP, 1979.

Eisenstein, Elizabeth. "Some Conjectures About the Impact of Printing on Western Society and Thought: A Preliminary Report." The Journal of Modern History 4.1 (March 1968): 1-56.

Elbow, Peter. Writing Without Teachers. New York: Oxford UP, 1973.

Febrve, Lucien and Henri-Jean Martin. The Coming of the Book: The Impact of Printing 1450-1800. Trans. David Gerard. Ed. Geoffrey Nowell-Smith and David Wooton. 1958. London: NLB, 1976.

Fish, Stanley. Is There a Text in This Class? The Authority of Interpretive Communities. Cambridge: Harvard UP, 1980.

Flower, Linda and John R. Hayes. "The Cognition of Discovery: Defining a Rhetorical Problem." College Composition and Communication 31.1 (Feb. 1980): 21-32.

Focus: Teaching English Language Arts 9.3 (Spring 1983). Special Issue: "Computers in Teaching English." Available through the NCTE Publications Department.

Foucault, Michel. An Introduction. Vol. 1 of The History of Sexuality. Trans. Robert Hurley. New York: Random House, 1980.

Gage, John T. "Philosophies of Style and Their Implications for Composition." College English 41.6 (Feb. 1980): 615-622.

Gonzales, Laurence. Computers for Writers. New York: Ballantine, 1984.

"A Guide to Personal Computing." Advertising Supplement. New York Times 3 June 1984.

Gula, Robert J. "Beyond the Typewriter: An English Teacher Looks at the Word Processor." Classroom Computer News May/June 1982: 31-32.

Hartman, Geoffrey. Criticism in The Wildnerness: The Study of Literature Today. New Haven: Yale UP, 1980.

Haselkorn, Mark P. and Jim S. Borck. "The Word Processor and the Scholarly Journal." McKenzie, ed. 49-58.

Heilbrun, Carolyn. "Some Words for The Trepid." McKenzie, ed. 1-9.

Hubbard, Francis A. Rev. of Writing in the Computer Age, et al. College English 46.2 (Feb. 1984): 128-133.

Jobst, Jack. "Computers and the Obsolete English Teacher." Focus 89-92.

Keogh, Jim. "Software Fit for a Hemingway." Personal Software Dec. 1983: 168-183.

Kinneavy, James. "The Relation of The Whole to the Part in Interpretration Theory." Linguistics, Stylistics, and the Teaching of Writing. Ed. Donald McQuade. Akron, OH: University of Akron P, 1979. 1-23.

_____. "Restoring the Humanities: The Return of Rhetoric From Exile." The Rhetorical Tradition and Modern Writing. Ed. James J. Murphy. New York: Modern Language Association, 1982. 19-28.

Kreiter-Kurylo, Carolyn. "Computers and Composition." The Writing Instructor 2.4 (1983): 174-181.

Lathrop, Ann. "Course-Ware Selection." Lawlor 47-53.

Lawlor, Joseph, ed. Computers in Composition Instruction. Los Alamitos, CA: SWRL Educational Research and Development, 1982.

Lederman, Marie Jean. "Preface." Akst 1.

_____, et al. Assessment and Improvement of the Academic Skills of Entering Freshman Students: A National Survey. CUNY Research Monograph Series: Report No. 5. New York: Instructional Resource Center, CUNY Office of Academic Affairs, 1983.

Marcus, Steven. "Real-Time Gadgets with Feedback: Special Effects in Computer-Assisted Instruction." The Writing Instructor 2.4 (1983): 156-164.

Marx, Karl and Frederick Engels. The German Ideology, Part I. Ed. C.J. Arthur. New York: International Publishers, 1970.

McAllister, Carol. "Visual Aid for the Writing Teacher: Into the Classroom With a Word Processor." Akst 27-29.

McKenzie, Alan T. "Introduction." McKenzie, ed. ix-xii.

_____. "The Word Processor and the Typing Chores of the Academic Department." McKenzie, ed. 25-28.

_____. "The Word Processor and the Work Habits of the Humanists." McKenzie, ed. 10-24.

_____, ed. A Grin on the Interface: Word Processing for the Academic Humanist. New York: Modern Language Association, 1984.

McWilliams, Peter. The Word Processing Book: A Short Course in Computer Literacy. Los Angeles: Prelude, 1983.

Moran, Charles. "Word Processing and the Teaching of Writing." English Journal 72 (March 1983): 113-115.

Mulder, Thomas. "Resources on Computer Use in English Instruction." Focus 135-144.

Naiman, Arthur. Introduction to Wordstar. Berkeley, CA: Sybex, 1982.

_____. Word Processing Buyer's Guide. Peterborough, NH: BYTE/McGraw-Hill, 1983.

Oates, William R. "The Importance of the Teacher in Computer-Assisted Basic Writing Skills Review." Akst 32-34.

Ong, Walter. "Oral Culture and the Literate Mind." Minority Language and Literature. Ed. Dexter Fisher. New York: Modern Language Association, 1977.

Reinecke, Ian. Microcomputers. Ringwood, Australia: Penguin, 1983.

Sandberg-Diment, Erik. "But Is It Writing?" New York Times 26 June 1984: C5.

Santayana, George. Character and Opinion in the United States. 1920. New York: George Braziller, 1955.

Schmeltz, L.R. Wordprocessing with Your Microcomputer. Blue Ridge Summit, PA: Tab Books, 1982.

Scholes, Robert. Semiotics and Interpretation. New Haven: Yale UP, 1982.

_____ and Nancy Comley. The Practice of Writing. New York: St. Martin's, 1981.

Schwartz, Helen J. "Monsters and Mentors: Computer Applications for Humanistic Education. College English 44.2 (Feb. 1982): 141-152.

_____. "Teaching Writing with Computer Aids." College English 46.3 (March 1984): 239-247.

Schwartz, Mimi. "Computers and the Teaching of Writing." Educational Technology 22 (Nov. 1982): 27-29.

Self, Warren. "Computers: Changing Writers' Relationships with Their Composing Processes." Focus 17-21.

Selfe, Cynthia A. and Billie J. Wahlstrom. "The Benevolent Beast: Computer-Assisted Instruction for the Teaching of Writing." The Writing Instructor 2.4 (1983): 183-192.

Shaughnessy, Mina. Errors and Expectations: A Guide for the Teacher of Basic Writing. New York: Oxford UP, 1977.

Sims, Carol. "Conference Report: Computers and the Contemporary Scholar." Clarion 13.4 (Dec. 1983): 6-7.

Sommer, Elyse. Perfect Writer Made Perfectly Clear. Radnor, PA: Chilton, 1984.

Sommers, Elizabeth Ann. "A Writing Teacher Experiments with Word Processing." Computers and Composition 1.2 (Feb. 1984): 1-3.

Southwell, Michael G. "Computer-Assisted Instruction in Composition at York College/CUNY: Grammar for Basic Writing Students." The Writing Instructor 2.4 (1983): 165-173.

Standiford, Sally N., et al. Computers in the English Classroom: A Primer for Teachers. Urbana, IL: Eric/NCTE, 1983.

Townsend, Carl. Exploring Word Processors: CP/M Edition. Beaverton, OR: Dilithium Press, 1984.

Van Nostrand, A.D., et al. Functional Writing. Boston: Houghton Mifflin, 1978.

Vogeli, Bruce. "Redefining Basic Skills in a Computer Culture." Akst 5-7, 86-90.

Williams, Raymond. Culture and Society: 1780-1950. 1958. New York: Columbia UP, 1983.

Wresch, William. "Computers and Composition Instruction: An Update." College English 45.8 (Dec. 1983): 794-799.

Zinsser. William. Writing With a Word Processor. New York: Harper and Row, 1983.